Ontology-Based Information Sharing in Weakly Structured Environments

Stuckenschmidt, H.

2003

document version
Publisher's PDF, also known as Version of record

VRIJE UNIVERSITEIT

Ontology-Based Information Sharing in Weakly Structured Environments

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor aan
de Vrije Universiteit Amsterdam,
op gezag van de rector magnificus
prof.dr. T. Sminia,
in het openbaar te verdedigen
ten overstaan van de promotiecommissie
van de faculteit der Exacte Wetenschappen
op donderdag 23 januari 2003 om 13.45 uur
in het auditorium van de universiteit,
De Boelelaan 1105

door

Heiner Stuckenschmidt

geboren te Bremen, Duitsland

promotor: prof.dr. F.A.H van Harmelen
copromotor: prof.dr. O. Herzog

Acknowledgements

I would like to thank people that supported me during the process of writing this thesis. First of all I have to mention my supervisors Frank van Harmelen and Otthein Herzog, Prof. Herzog for giving me complete freedom and support in my research and enabling me to come up with and work on many different ideas and Frank van Harmelen for forcing me to bring all these ideas together again in terms of a coherent document.

I also want to thank all the people I worked together with during the last three years. Many of the initial ideas in this work have been developed during my work in the BUSTER project at the University of Bremen. I had fruitful discussions and collaboration with the colleagues and students in the project. I want to mention two people in particular: Ubbo Visser who came up with the interesting problem of semantic interoperability and Holger Wache, who was the first to recognize the theoretical issues behind the problem.

Later, my work very much profited from working together with Frank and Harmelen and Dieter Fensel in Amsterdam who fed me with the idea of the Semantic Web and had a great influence on the further development of my research and with Jerome Euzenat in Grenoble where I made important progress on the topic of language interoperability.

Maybe the most exciting part of the research was working together with my Master's Students Jens Hartmann and Gerhard Schuster who made valuable contributions to questions addressed in this thesis.

Further I am grateful to Michel Klein for translating the summary of the thesis and Ingo Timm for many meta-level discussions about sense and nonsense of writing a PhD.

Last but not least I would thank Maike Carstens for proof-reading as well as together with my parents and all my other friends for accepting my style of living and providing me with the kind of environment I needed to get computer science out of my head from time to time.

SIKS Dissertation Series No. 2003-01

The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

Contents

Chapter 1

Introduction

The so-called information society demands for complete access to available information, which is often heterogeneous and distributed. Most information systems use specific data models and databases for this purpose. This implies that making new data available to a system requires, that the data is either transferred into the system's specific data format or is even acquired again. This process is very time consuming and tedious. Data acquisition, automatically or semi-automatically, often makes large-scale investment in technical infrastructure and/or manpower inevitable. The idea of information sharing is to overcome these problems by providing common access to existing information sources The advantages of successful information sharing are obvious for many reasons:

- Quality improvement of data due to the availability of large and more complete data sets.
- Cost reduction resulting from multiple use of the existing information sources.
- Avoidance of redundant data and conflicts that can arise from redundancy.

However, in order to establish efficient information sharing, many technical problems have to be solved. Firstly, a suitable information source must be located that might contain data needed for a given task. Finding suitable information sources is a problem addressed in the areas of information retrieval [van Rijsbergen, 1979, Frakes and Baeza-Yates, 1992] and information filtering [Sheth and Maes, 1993, Belkin and Croft, 1992] and will not be addressed in this work. In the following we always assume that we know the information sources, e.g. web sites, whose content we want to share.

Once the information source has been found, access to the data therein has to be provided. This means that each of the information sources found in the first step have to work together with the system that is using the information.

The problem of bringing together heterogeneous and distributed computer systems is known as *interoperability*. Interoperability has to be provided on a technical and informational level. In short, information sharing not only needs to provide full accessibility to the data, it also requires that the accessed data can be processed and interpreted by the remote system.

It has been argued that ontologies are a key technology for this kind of applications [Fensel, 2001, Uschold and Gruninger, 1996, Gruninger and Uschold, 2002]. Successful approaches and applications are reported from the database area where ontologies have been used to enable *Intelligent Information Integration* (e.g. [Arens et al., 1993, Kashyap and Sheth, 1998, Levy et al., 1996, Calvanese et al., 1998b, Preece et al., 1999]). However, many of these approaches rely on the existence of well-established data structures that can be used to analyze and exchange information. There are attempts to extend existing approaches to semi-structured information. However handling less structured domains such as the World Wide Web is still an unsolved problems. In this work we will discuss the following thesis:

Thesis 1 (Ontology-Based Information Sharing) *: Ontologies are a suitable technology for information sharing in weakly structured information environments.*

In order to be able to defend this thesis, we first have to define the key notion used in the thesis in more detail:

Definition 1.1 (Weakly-Structured Environment) *A weakly structured information environment is a collection of information items that are characterized by:*

1. *the lack of precisely defined conceptual and logical data-models.*

2. *the lack of clear system boundaries due to dynamic addition and removal of information.*

3. *the lack of a unique representation for information semantics.*

These characteristics distinguishes this thesis from previous work on information sharing and integration which is concerned with environments where at least one, but often more of these characteristics are not taken into account.

In order to establish our thesis, several questions have to be answered that will be discussed in the next section.

1.1 Goal and Research Questions

The goal of this work is to contribute to the development of a framework that covers the whole process of representing, deploying and using ontologies for information sharing in a distributed and heterogeneous environment such as the World Wide Web. In order to establish a framework for ontology-based information sharing, we have to answer some fundamental questions:

A: **How do we represent and reason about ontologies ?**
We need to be able to write down ontologies in a way that enables systems to process them. This is essential, as we want to automate parts of the information sharing process. We have to think about ontology representations, that are compatible with existing standards on the World Wide Web that have a clear semantics and efficient reasoning support. At the same time these languages have to be able to capture all the knowledge we need to share information.

B: **What kind of ontologies are needed and how to build them ?**
Ontologies can be used in very different ways both in general [Jasper and Uschold, 1999] and for information integration [Wache et al., 2001]. We have to define an architecture that balances the use of ontologies as global references and as carriers of local semantics. Equally important is the provision of guidance for installing this architecture. More specifically, we have to support ontology construction with respect to the ontologies used in the framework.

C: **How can we connect ontologies and information ?**
Our ultimate goal is the exchange of information. Ontologies are just a means for reaching this goal. In order to make use of them, we have to establish a connection between the ontologies and the concrete information we want to exchange. This connection has to be tight enough to facilitate information sharing on a semantic level. On the other hand, we have to provide strong support for this task in order to overcome the annotation bottleneck.

D: **How can we use ontologies in order to improve semantic interoperability ?** A weak structuring of information implies a weak connection between information content and ontologies used. Further, in a heterogeneous and distributed environment we will have to deal with more than one ontology. These characteristics raise the question, whether ontologies can still be used in order to improve information sharing. Conventional reasoning techniques fail to provide enough support. We therefore have to think about new methods and their use for finding and accessing information.

In the remainder of this work we will discuss these questions and thereby build up a framework for supporting information sharing in weakly structured

environments, i.e. the World Wide Web. We will base the framework on existing work and combine or supplement it by new methods whenever necessary.

1.2 Contributions to the State of the Art

The main contribution of this thesis is preparing the ground for a framework that covers the complete process of using ontologies for information sharing from the representation of ontologies and their deployment in a heterogeneous environment to their use for information filtering and exchange. There are some points in this process with more specific contributions:

A: **Ontology Language Interoperability**
The current approach to ensure interoperability is to define one standard language everybody has to commit to. We describe a new approach to ontology language interoperability that is more flexible than existing approaches, but still guarantees certain formal properties. We introduce the general idea of the approach and investigate it in more details for the special case of terminological languages, which are of interest for encoding ontologies.

B: **Ontology Infrastructure Development**
At the moment, most information sharing applications on the web assume the existence of a global ontology (compare, e.g., [Fensel et al., 1998] or [Guarino et al., 1999]). New ontologies are either merged with the existing ones forming an even bigger global ontology or they have to be connected with all previously existing ontologies. Both approaches cause problems. Global ontologies are hard to maintain and extend, mappings between local ontologies are hard to find and the effort of building mappings grows with every new ontology. We propose an ontology infrastructure that trades off local and global ontologies in such a way that systems can define terms individually whose meaning can still be compared across systems without explicit mappings.

C: **Metadata Creation and Management**
One of the most challenging problems of using ontologies on the World Wide Web is the need to relate huge amounts of information to these ontologies. This problem has been addressed in different ways including the application of machine learning techniques [Jenkins et al., 1999] and the development of advanced annotation tools [Staab et al., 2001]. However, much of the work of relating information to ontologies is still left to the user. We present a method that combines different editing, reasoning and learning tools and methods in such a way that the generation and validation of content-related metadata can be automated to a large extend making use of the little structure available on the web.

D: **Semantic Mapping and Content-Based Retrieval**
Current work on using ontologies for information retrieval and seman-

tic mapping on the one and terminological reasoning on the other hand is mostly done separately from each other in semantic web research[1]. Terminological reasoning is mostly used at ontology development time in order to check consistency and to find hidden subclass relations. The main reason for this is that terminological reasoners depend on a complete and homogeneous model and can only reveal exact matches based on logical proofs. We adapt and extend terminological reasoning methods in such a way that they can be used to reason across different heterogeneous ontologies by computing correct approximations of the intended answer. This bridges the gap between work on semantic mapping and on terminological reasoning.

We will present these contributions successively while describing the overall framework. In the next section we will give an outline of the thesis that clarifies the relation between the chosen structure and the goals and contributions mentioned above.

1.3 Overview of the Work

This thesis is organized in three main parts each one covering a specific aspect of the overall problem information sharing in weakly structured environments:

Part I: Semantic Interoperability In the first part of the thesis we will discuss languages for representing explicit models of information semantics and the problem of achieving interoperability between these languages which is a prerequisite for a combined use of semantic models.

Part II: Ontologies and Metadata The second part of the thesis is concerned with the problem of how to generate semantic descriptions of information using the languages considered in the first part. We further discuss how information semantics can be connected with information using metadata.

Part III: Information Sharing In the third part of the thesis we discuss methods for information filtering and integration that work on explicit models of information semantics in terms of ontologies and metadata. After a theoretical discussion we describe an implementation of these methods in an intelligent information sharing solution.

We summarize with a discussion of contributions to the state of the art provided by this thesis and give an outline of an agenda for future research on the overall topic of ontology-based information sharing. The work is organized in chapters as follows.

[1] An exception is [Calvanese et al., 2001]

Chapter 2 is an extended motivation for the goals of this thesis. We discuss the problem of information sharing and point out existing work we build upon. We further argue for the need of an investigation on the semantic level and clarify why ontologies are chosen as the basic technology.

Part I

Chapter 3 prepares the ground for research question A by reviewing current proposals of languages for encoding ontologies on the Web, namely RDF schemas and DAML+OIL. We further give an overview over existing proposals for web-based ontology languages and argue that there is a need for language level integration.

Chapter 4 directly addresses question A by providing a formal investigation of a flexible framework for defining ontology languages and translating between them in such a way that results of logical reasoning are preserved in the transformed model. We also describe how the flexible language scheme can be implemented using existing Web technology, i.e. XML and XSLT.

Part II

Chapter 5 moves on to question B. Assuming that there is a language for encoding ontologies in a common semantic framework, in this chapter the question of a suitable architecture is raised. We sketch the process of exchanging meaning between systems and derive an ontology infrastructure that combines the use of local and global ontologies. We also introduce a specialized strategy for building this infrastructure.

Chapter 6 is concerned with the relation between the ontological infrastructure defined before and information presented on the web. We discuss the role of metadata as providing the glue between weakly structured information and ontologies. The rest of the chapter describes a method for almost automatically generating metadata descriptions for HTML documents that links web pages to ontological classes. We report successful experiments we conducted on a Web-based information system.

Part III

Chapter 7 finally addresses the problem of using the infrastructure in terms of ontologies encoded in a common semantic framework and metadata descriptions assigning web pages to these ontologies for information sharing (question D). We briefly review the notion semantics preserving translations proposed by [Ciocoiu and Nau, 2000]. Based on this notion we describe methods for reclassifying information across systems and show that these methods can be used to locate interesting information across heterogeneous systems based on their content.

Chapter 8 is meant to demonstrate the practical value of the models and methods developed in the thesis. We do this by describing the prototype of an intelligent middle-ware for information sharing that implements many of the ideas described throughout this thesis. We especially pay attention to those aspects of the prototype that are directly related to our results, namely the search and translation functions that directly correspond to the methods described in chapter 7.

Chapter 9 summarizes the results of the thesis. We collect and explain the most important insights and design decisions that influenced the framework described in this thesis. We also review the work in relation to our goals and research questions and draw conclusions on the contribution of the work to providing answers to these questions. Finally, we point out to future work that may extend and improve the results presented.

Chapter 2

Integration, Semantics and Ontologies

The goal of this chapter is to give extended motivation for the research questions raised in the introduction. We address the problem of heterogeneity and argue that explicit representations of information semantics are needed in a weakly structured environment. In order to support this claim, we give a hypothetical application example illustrating the benefits of explicit semantics. Further, we introduce ontologies as a fundamental technology for explicating semantics that will be in the focus of discussions in the thesis.

Acknowledgements: Parts of this chapter have been published before, the description of syntactic approaches to information sharing is taken from [Visser et al., 2001]. The description of ontologies and their applications appeared in [Visser et al., 2002]. The former is co-authored by Ubbo Visser, Holger Wache and Thomas Voegele, the latter by Ubbo Visser, Gerhard Schuster and Thomas Voegele.

The problem of providing access to information has been largely solved by the invention of large-scale computer networks (i.e. the World Wide Web). The problem of processing and interpreting retrieved information, however, remains an important research topic called Intelligent Information Integration [Fensel, 1999, Wiederhold, 1996]. Problems that might arise due to heterogeneity of the data are already well known within the distributed database systems community (e. g. [Kim and Seo, 1991], [Kashyap and Sheth, 1998]). In general, heterogeneity problems can be divided into three categories:

1. Syntax (e. g. data format heterogeneity),
2. Structure (e. g. homonyms, synonyms or different attributes in database tables), and

3. Semantics (e. g. intended meaning of terms in a special context or application).

Throughout this thesis we will focus on the problem of semantic integration and content-based filtering, because sophisticated solutions to syntactic and structural problems have been developed. On the syntactical level, standardization is an important topic. Many standards have evolved that can be used to integrate different information sources. Beside the classical database interfaces like ODBC, web-oriented standards like HTML [Ragget et al., 1999], XML [Bray et al., 1998] and RDF [Lassila and Swick, 1999] gain importance (see http://www.w3c.org). As the World Wide Web offers the greatest potential for sharing information, we will base our work on these evolving standards that will be briefly introduced in the next section.

2.1 Syntactic Standards

Due to the extended use of computer networks, standard languages proposed by the W3C committee are rapidly gaining importance. Some of these standards are reviewed in the context of information sharing. Our main focus is on the extensible markup language XML and the resource description format RDF, However, we briefly discuss the hypertext markup language for motivation.

2.1.1 HTML: Visualizing Information

Creating a web page on the Internet was the first, and currently the most frequently and extensively used technique for sharing information. These pages contain information with both free and structured text, images and possibly audio and video sequences. The hypertext markup language is used to create these pages. The language provides primitives called tags that can be used to annotate text or embedded files in order to determine the order in which they should be visualized. The tags have a uniform syntax enabling browsers to identify them as layout information when parsing a page and generating the layout:

```
<tag-name> information (free text) </tag-name>
```

It is important to note that the markup provided by HTML does not refer to the content of the information provided, but only covers the way it should be structured and presented on the page. On one hand, this restriction of visual features is a big advantage, because it enables us to share highly heterogeneous knowledge, namely arbitrary compositions of natural language texts and digital media. On the other hand, it is a big disadvantage, because the process of understanding the content and assessing its value for a given task is mostly left to the user.

HTML was created to make information processable by machines, but not understandable. The conception of HTML, offering freedom of saying anything about any subject, led to a wide acceptance of the new technology. However, the internet has a most challenging problem, its inherent heterogeneity. One way to cope with this problem appears to be an extensive use of support technology for browsing, searching and filtering of information based on techniques that do not rely on fixed structures. In order to build systems that support access to this information we have to find ways to handle the heterogeneity without reducing the "freedom" too much. This is accomplished by providing machine-readable and/or machine understandable information about the content of a web page.

2.1.2 XML: Exchanging Information

In order to overcome the fixed annotation scheme provided by HTML that does not allow to define data structures, XML was proposed as an extensible language allowing the user to define his own tags in order to indicate the type of content annotated by the tag. First intended for defining document structures in the spirit of the SGML document definition language [ISO-8879, 1986] (XML is a subset of SGML) it turned out that the main benefit of XML actually lies in the opportunity to exchange data in a structured way. Recently XML schema were introduced [Fallside, 2000] that could be seen as a definition language for data structures emphasizing this idea. In the following we sketch the idea behind XML and describe XML schema definitions and their potential use for data exchange.

A data object is said to be an XML document if it follows the guidelines for well-formed documents provided by the W3C committee. The specification provides a formal grammar used in well-formed documents. In addition to general grammar, the user can impose further grammatical constraints on the structure of a document using a document type definition (DTD). An XML document is then valid if it has an associated type definition and complies with the grammatical constraints of that definition. A DTD specifies elements that can be used within a XML document. In the document, the elements are delimited by start and end tags. Furthermore, it has a type and may have a set of attribute specifications consisting of a name and a value.

The additional constraints in a document type definition refer to the logical structure of the document, this specifically includes the nesting of tags inside the information body that is allowed and/or required. Further restrictions that can be expressed in a document-type definition concern the type of attributes and the default values to be used when no attribute value is provided. At this point, we ignore the original way DTDs are defined, because XML schemas, which are described next, provide a much more comprehensible way of defining the structure of an XML document.

An XML schema is itself an XML document defining the valid structure of an

XML documents in the spirit of a DTD. The elements used in a schema definition are of the type 'element' and have attributes that define the restrictions already mentioned. The information within such an element is simply a list of further element definitions that have to be nested inside the defined element:

```
<element name="value" type="value" ...>
    <element name="value" minOccurs="value" ... />
        ...
</element>
```

Additionally, XML schema have other features that are very useful for defining data structures:

- Support for basic data types [Thompson et al., 2000]
- Constraints on attributes (e.g.; occurrence constraints)
- Sophisticated structures [Biron and Malhotra, 2000] (e.g.; definitions derived by extending or restricting other definitions)
- A name-space mechanism allowing the combination of different schemas

We will not be discussing these features in detail. However, it should be mentioned that the additional features make it possible to encode rather complex data structures. This enables us to map the data-models of applications, whose information we wish to share with others on an XML schema [Decker et al., 2000]. Once mapped, we can encode our information in terms of an XML document and make it (combined with the XML schema document) available over the Internet. The exchange of information mediated across different formats in the following way:

$$\text{Application Data Model} \leftrightarrow \text{XML schema} \rightarrow \text{XML document}$$

This method has great potential for the actual exchange of data. However, the user must commit to our data-model in order to make use of the information. As subsequently and previously mentioned, an XML schema defines the structure of data and provides no information about the content or the potential use of the information. Therefore, it lacks an important advantage of meta-information, which is now discussed in the next section.

2.1.3 RDF: A Data-Model for Meta-Information

Previously, we stated that XML is designed to provide an interchange format for weakly structured data by defining the underlying data model in a schema and using annotations from the schema in order to relate information items to the schema specification. We have to notice that:

- XML is purely syntactic/structural in nature
- XML describes data on the object level
- XML often encodes an application-specific data model

Consequently, we have to look for further approaches if we want to describe information on the meta level and define its meaning. In order to fill this gap, the RDF standard has been proposed as a data model for representing meta-data about web pages and their content using XML syntax.

The basic model underlying RDF is very simple. Every type of information about a resource, which may be a web page or an XML element, is expressed in terms of a triple:

```
(resource, property, value)
```

Thereby, the property is a two-placed relation that connects a resource to a certain value of that property. A value can be a simple data type or a resource. Additionally, the value can be replaced by a variable representing a resource that is further described by linking triples making assertions about the properties of the resource that is represented by the variable:

```
(resource, property, X)
    (X, property_1, value_1)
        ...
    (X, property_n, value_n)
```

Another feature of RDF is its reification mechanism that makes it possible to use an RDF-triple as value for the property of a resource. Using the reification mechanism we can make statements about facts. Reification is expressed by nesting triples:

```
(resource_1, propery_1, (resource_2, property_2, value))
```

Further, RDF allows multiple values for single properties. For this purpose, the model contains three built-in data-types called collections, namely unordered lists (bag) ordered lists (seq) and sets of alternatives (alt) providing some kind of an aggregation mechanism.

A further problem arising from the nature of the Web is the need to avoid name-clashes that might occur when referring to different web-sites that might use different RDF-models to annotate meta-data. RDF uses name-spaces that are provided by XML in order to overcome this problem. They are defined once by referring to a URI that provides the names and connects it to a source-ID that is then used to annotate each name in an RDF specification defining the origin of that particular name:

```
source_id:name
```

A standard syntax has been defined to write down RDF statements making it possible to identify the statements as meta-data. Thereby providing a low level language for expressing the intended meaning of information in a machine processable way.

2.2 Structure Integration

The first problem that goes beyond the purely syntactic level is the integration of heterogeneous structures. For a long time, this problem had to be solved by manually coded transformation queries. Recently there has been an increased interest in more sophisticated middle-ware components that enable the user to define flexible mapping relations between different data models. These so-called mediator systems defining mapping rules between different information structures [Wiederhold, 1992] provide an elegant solution for the structural integration problem. In the following we will quickly review some existing solutions for the problem of information integration on a syntactic and structural level, respectively. Throughout this thesis we will consider these approaches as being given and build our own work on top of these technologies. We will discuss single aspects of these approaches in more detail whenever it is necessary.

The problem of integrating heterogeneous database schemas has been addressed by many researchers (see [Levy, 1999] or [Wache et al., 2001] for surveys). The results achieved in this area can be used to integrate information at a structural level, i.e. to integrate different XML models. In the following, we briefly introduce the basic technologies and argue why they are not sufficient for weakly structured environment in the sense of definition 1.1.

2.2.1 Schemas, Views and Queries

A schema is normally seen as a set of named relations. The positions in the relations also called columns are named and considered part of the schema. The typical way of accessing information is in terms of conjunctive queries of the form:

$$q(\bar{X}) \Leftarrow e_1(\bar{X}_1) \wedge \cdots \wedge e_n(\bar{X}_n)$$

where $e_1, \cdots, e_n$ are relations and $\bar{X}_1, \cdots, \bar{X}_n$ are element tuples of these relations. The result of the query is a set of tuples satisfying $q(\bar{X})$. Multiple queries with the same head clause can be used to drive the union of individual schemas. A view is a named query.

The integration of heterogeneous schemas is normally done using a global schema that is connected to the heterogeneous schemas to be integrated by a number of views. We can distinguish two general approaches:

Global-As-View: In the global-as-view approach every relation in the global schema is defined as a view over the different schemas to be integrated (see e.g. [Garcia-Molina et al., 1997]. In especially, the integration is defined by horn rules where the consequence is a relation in the integrated schema and the clauses in the antecedents correspond to relations in the different heterogeneous schemas. Queries to the global schema can easily be answered by unfolding the clauses making use of the view definitions. The unfolded query can be computed using conventional techniques used in database systems. A drawback of this approach is that the independence of the individual information systems is lost due to their combined use in queries. This leads to problems if information sources are added or removed from the integrated system.

Local-As-View: In the local-as-view approach, views are used in the opposite way (see for example [Levy et al., 1996]). In order to connect local schemas with the integrated one views of the following form are used:

$$s_i : e_j(\bar{X}_j) \Leftarrow q_1(\bar{X}_1) \wedge \cdots \wedge q_n(\bar{X}_n)$$

This means that single relations in the schemas to be integrated are used to define more complex information structures in the integrated schema. This also means that additional properties of the integrated information to be included in the global schema can be encoded in the mapping rules. As every local schema is defined as a view over the global one, the independence of the individual sources are preserved. The drawback of this approach is that answering queries over the integrated view is more difficult as it corresponds to the problem of abduction.

2.2.2 Possibilities and Limitations of Schema-Based Approaches

A natural question that arises in connection with work on information integration is how far the approaches developed already solve the problem of information sharing in weakly structured environments. It has been shown that results from the database area provide partial solution for the integration semi-structured information, i.e. of XML documents, because an XML encoding can be seen as a schema that provides the basis for applying one of the approaches described above (see e.g. [Bergamaschi et al., 1999]).

Problems that remain are connected with the semantics of information that is not part of the XML schema specification. The reason for these problems is that current approaches from the database area only provide limited possibilities to map information semantics across systems. The global-as-view approach for example simply maps information structures of the global schema to structures in the local schema without being able to ship additional information about the tuples that are returned as a result. The local-as-view approach

improves this situation by providing the possibility to supplement the result of a query with additional information about the returned tuples. This additional information can be encoded in the conjuncts in the body of the view definition. They can for example be used to return contextual parameters of the individual source, such as the genre of a movie or the time period the information refers to (compare [Levy, 1999]).

In their basic form, none of the approaches considers the necessity of transforming the individual information items that are returned by a query. Such transformation might be necessary in order to cope with different scales or measures used to encode information in the different information sources. The problem of performing such translations is referred to as context transformation and has been discussed in some recent approaches [Goh et al., 1994, Wache and Stuckenschmidt, 2001]. While these approaches solve the context transformation problem for the case of database integration by defining so-called context transformation rules, they do not apply to weakly structured environments, because all existing approaches rely on an explicit notion of context given by the specific position in the schema. In database system the context of a specific information item is often not only given by the schema but also further defined in a Data Dictionary explaining its relation to other data elements and defining the basis for transformation in terms of the data type, its valid ranges and further constraints. Early approaches to database integration like Mermaid [Templeton et al., 1987] even directly used Data Dictionaries for accessing information.

In the absence of a well-defined data model, context transformation rules do not apply, as source and goal context are not defined, this in turn makes it impossible to determine the kind of transformation to be applied. In order to transform information items from one source to another, we have to rely on a different notion of context that does not refer to a schema. In the same way we have to establish ways of describing the context of information that play the same role as data dictionaries without being tight to the conceptual of logical data model.

2.3 Handling Information Semantics

In the following, we use the term *semantic integration* or *semantic translation*, to denote the resolution of semantic conflicts that occur between heterogeneous information systems in order to achieve semantic interoperability. For this purpose, the systems have to agree on the *meaning* of the information that is interchanged. Semantic conflicts occur whenever two systems do not use the same interpretation of the information. The simplest form of disagreement in the interpretation of information are homonyms (the use of the same word with different meanings) and synonyms (the use of different words with the same

meaning). However, these problems can be solved by one-to-one structural mappings. Therefore, most existing converter and mediator systems are able to solve semantic conflicts of this type. More interesting are conflicts where one-to-one mappings do not apply. In this case, the semantics of information has to be taken into account in order to decide how different information items relate to each other. Many attempts have been made in order to access information semantics. We will discuss general approaches to this problem with respect to information sharing.

2.3.1 Semantics from Structure

A common approach to capture information semantics is in terms of its structure. The use of conceptual models of stored information has a long tradition in database research. The most well-known approach is the Entity-Relationship approach [Chen, 1976]. Such conceptual models normally have a tight connection to the way the actual information is stored, because they are mainly used to structure information about complex domains. This connection has significant advantages for information sharing, because the conceptual model helps to access and validate information. The access to structured information resources can be provided by wrappers derived from the conceptual model [Wiederhold, 1992]. In the presence of less structured information sources, e.g. HTML pages on the web, the problem of accessing information is harder to solve. Recently, this problem has been successfully tackled by approaches that use machine learning techniques for inducing wrappers for less structured information. One of the most prominent approaches is reported in [Freitag and Kushmerick, 2000]. The result of the learning process is a set of extraction rules that can be used to extract information from web resources and insert it into a newly created structure that is used as a basis for further processing.

While wrapper induction provides a solution for the problem of extracting information from weakly structured resources, the problem of integrating information from different sources remains largely unsolved because extraction rules are solely defined on the structural level. In order to achieve an integration on the semantic level as well, a logical model has to be built on top of the information structure. We find two different approaches in literature.

Structure Resemblance: A logical model is built that is a a one-to-one copy of the conceptual structure of the database and encode it in a language that makes automated reasoning possible. The integration is then performed on the copy of the model and can easily be tracked back to the original data. This approach is implemented in the SIMS mediator [Arens et al., 1993] and also by the TSIMMIS system [Garcia-Molina et al., 1995]. A suitable encoding of the information structure can already be used in order to generate hypotheses about semantically related structures in two information sources.

Structure Enrichment: A logical model is built that resembles the structure of the information source and contains additional definitions of concepts. A detailed discussion of this kind of mapping is given in [Kashyap and Sheth, 1996]. Systems that use structure enrichment for information integration are OBSERVER [Kashyap and Sheth, 1998], KRAFT [Preece et al., 1999], PICSEL [Goasdoue and Reynaud, 1999] and DWQ [Calvanese et al., 1998b]. While OBSERVER uses description logics for both structure resemblance and additional definitions, PICSEL and DWQ define the structure of the information by (typed) horn rules. Additional definitions of concepts mentioned in these rules are done by a description logic model. KRAFT does not commit to a specific definition scheme.

The approaches are based on the assumption that the structure of the information already carries some semantics in terms of the domain knowledge of the database designer. We therefore think that the derivation of semantics from information structures is not applicable in an environment where weakly structured information has to be handled, because in most cases a conceptual model is not available.

2.3.2 Semantics from Text

An alternative approach for extracting semantic information from the structure of information resources is the derivation of semantics from text. This approach is attractive on the World Wide Web, because huge amounts of free text resources are available. Substantial results in using natural language processing come from the area of information retrieval [Lewis, 1996]. Here the task of finding relevant information on a specific topic is tackled by indexing free-text documents with weighted terms that are related to their contents. There are different methods for matching user queries against these weighted terms. It has been shown that statistical methods outperform discrete methods [Salton, 1986]. As in this approach the semantics of a document is contained in the indexing terms their choice and generation is the crucial step in handling information semantics. Results of experiments have shown that document retrieval using stemmed natural language terms taken from a document for indexing it is comparable to the use of controlled languages [Turtle and Croft, 1991]. However, it is argued that the use of compound expressions or propositional statements (very similar to RDF) will increase precision and recall [Lewis, 1996].

The crucial task in using natural language as a source of semantic information is the analysis of documents and the generation of indexing descriptions from the document text. Straightforward approaches based on the number of occurrences of a term in the document suffer from the problem that the same term may be used in different ways. The same word may be used as a verb or as an adjective (*fabricated units* vs. *they fabricated units*) leading to different degrees of relevance with respect to a user query. Recent work has shown that retrieval results can be improved by making the role of a term

in a text explicit [Basili et al., 2001]. Further, the same natural language term may have different meanings even within the same text. The task of determining the intended meaning is referred to as word-sense disambiguation. A prominent approach is to analyze the context of a term under consideration and decide between different possible interpretations based on the occurrence of other words in this context that provide evidence for one meaning. The exploitation of these implicit structures referred to as latent semantic indexing [Deerwester et al., 1990]. The decision for a possible sense is often based on a general natural language thesaurus (see e.g. [Yarowsky, 1992]). In the case where specialized vocabularies are used in documents, explicit representations of relations between terms have to be used. These are provided by domain specific thesauri [Maynard and Ananiadou, 1998] or semantic networks [Gaizauskas and Humphreys, 1997]. Extracting more complex indexing information such as propositional statements is mostly unexplored. Ontologies, which will be discussed later, provide possibilities for using such expressive annotations.

Despite the progress made in natural language processing and the its successful application to information extraction and information retrieval, there are still many limitations due to the lack of explicit semantic information. While many ambiguities in natural language can be resolved by the use of contextual information, artificially invented terms cause problems, because their meaning can often not be deduced from every day language, but depends on the specific use of the information source. In this case we have to rely on the existence of corresponding background information. We will give examples for such situations in section 2.4.

2.3.3 The Need for Explicit Semantics

In the last section we reviewed approaches for capturing information semantics. We concluded that the derivation of semantics from structures does not easily apply to weakly structured information. The alternative of using text understanding techniques on the other hand works quite well for textual information that contains terms from every day language, for in this case existing linguistic resources can be used to disambiguate the meaning of single words. The extraction of more complex indexing expressions is less well investigated. Such indexing terms, however, can be easily derived from explicit models of information semantics. A second shortcoming of approaches that purely rely on the extraction of semantics from texts is the ability to handle special terminology as it is used by scientific communities or technical disciplines.

The problems of the approaches mentioned above all originated from the lack of an explicit model of information semantics. Recently, The need for a partial explication of information semantics has been recognized in connection with the World Wide Web. Fensel identifies a three level solution to the problem of developing intelligent applications on the web [Fensel, 2001]:

Information Extraction: In order to provide access to information resources, information extraction techniques have to be applied providing wrapping technology for a uniform access to information.

Processable Semantics: Formal languages have to be developed that are able to capture information structures as well as meta-information about the nature of information and the conceptual structure underlying an information source.

Ontologies: The information sources have to be enriched with semantic information using the languages mentioned in step two. This semantic information has to be based on a vocabulary that reflects a consensual and formal specification of the conceptualization of the domain, also called an ontology.

The first layer directly corresponds to the approaches for accessing information discussed in the beginning of this section. The second layer partly corresponds to the use of the annotation languages XML and RDF mentioned in connection with the syntactic and structural approaches. The third layer, namely the enrichment of information sources with additional semantic information and the use of shared term definitions has already been implemented in recent approaches for information sharing in terms of meta-annotations and terms definitions. We would like to emphasize that the use of explicit semantics is no contradiction to the other approaches mentioned above. Using explicit models of information semantics is rather a technique to improve or enable the other approaches. However, we think that large scale information sharing requires explicit semantic models.

In information sources specialized vocabularies often occur in terms of classifications and assessments used to reduce the amount of data that has to be stored in an information source. Instead of describing all characteristics of an object represented by a data-set a single term is used that relates the object to a class of objects that share a certain set of properties. This term often corresponds to a classification that is specified outside the information source. The use of product categories in electronic commerce or the relation to a standard land-use classification in geographic information systems are examples for this phenomenon. A special kind of classification is the use of terms that represent the result of an assessment of the object described by the data-set. In e-commerce systems, for example, customers might be assigned to different target groups whereas the state of the environment is a typical kind of assessment stored in geographic information systems.

We believe that classifications and assessments, which can be seen as a special case of a classification, play a central role in large scale information sharing, because their ability to reduce the information load by abstracting from details provides means to handle very large information networks like the World Wide Web. Web directories like Yahoo! (`www.yahoo.com`) or the open directory

project (dmoz.org) organize millions of web pages according to a fixed classification hierarchy. Beyond this, significant success has been reached in the area of document and web page classification (see [Pierre, 2001] or [Boley et al., 1999]). Apart from the high relevance for information sharing on the World Wide Web, being able to cope with heterogeneous classification schemes is also relevant for information integration in general. In the following we give two examples of the use of specific classifications in conventional information systems and illustrate the role of explicit semantic models in providing interoperability between systems:

2.4 An Example: Water Quality Assessment

We will now describe a typical situation that addresses semantic aspects of information sharing. The example is simplified but it tries to give the general idea of situations where semantic integration is necessary and how it could look like. We assume that we have a database of measured toxin values for wells in a certain area. The database may contain various parameters. For the sake of simplicity we restrict our investigation to two categories each containing two parameters:

- Bacteria: Faecal Coliforms, Intestinal Helminth
- Salts: Sodium, Sulfate

Our scenario is concerned with the use of this information source for different purposes in environmental information systems. We consider two applications involving an assessment of the environmental impact. Both applications demand for a semantics-preserving transformation of the underlying data in order to get a correct assessment. While the first can be solved by simple mapping, the second transformation problem requires the full power of the classification based transformation described in the previous section, underlining the necessity for knowledge-based methods for semantic information integration.

2.4.1 Functional Transformation

A common feature of an environmental information system is the generation of geographic maps summarizing the state of the environment using different colors. High toxin values are normally indicated by a red color, low toxin values by a green color. If we want to generate such maps for the toxin categories 'Bacteria' and 'Salts' using the toxin database we have to perform a transformation on the data in order to move from sets of numerical values to discrete classes, in our case the classes 'red' and 'green'. If we neglect the problem of aggregating values from multiple measurements at the same well (this problem is addressed in [Keinitz, 1999]), this classification problem boils down to the application of a function that maps combinations of values on one of the discrete classes. The corresponding functions have to be defined by a

domain expert and could for example be represented by the tables shown below:

Faecal Coliforms	**Intestinal Helminth**	**Bacteria**
< 0.1	No	Green
< 0.1	Yes	Red
< 1.0	No	Green
< 1.0	Yes	Red
< 10.0	No	Green
< 10.0	Yes	Red
< 100.0	No	Red
< 100.0	Yes	Red
≥ 100.0	No	Red
≥ 100.0	Yes	Red

Sodium	**Sulfate**	**Salts Assess.**
< 0.1	< 1.0	Green
< 0.1	< 200.0	Green
< 0.1	< 300.0	Green
< 0.1	≥ 300.0	Red
< 20.0	< 1.0	Green
< 20.0	< 200.0	Green
< 20.0	< 300.0	Green
< 20.0	≥ 300.0	Red
< 200.0	< 1.0	Green
< 200.0	< 200.0	Green
< 200.0	< 300.0	Red
< 200.0	≥ 300.0	Red
≥ 200.0	< 1.0	Red
≥ 200.0	< 200.0	Red
≥ 200.0	< 300.0	Red
≥ 200.0	≥ 300.0	Red

This kind of semantic transformation can be performed by so-called context transformation rules [Wache, 1999]. These rules connect one or more database fields in the source database with a field in the target database (in our case the database of the environmental information system) and define a function that calculates the value to be inserted in the target field from the values in the source database.

2.4.2 Non-Functional Transformations

We have argued that simple rule-based transformations are not always sufficient for complex transformation tasks [Stuckenschmidt and Wache, 2000]. The need for more complex transformations becomes clear when we try to use the previously generated information to decide whether a well may be used for different purposes. We can think of three intended uses each with its own requirements on the pollution level that are assumed to be specified as follows:

Bathing: absence of Intestinal Helminth and a Faecal Coliform pollution that is below 12.0 mg/l

Drinking: absence of Intestinal Helminth, a Faecal Coliform pollution that is below 20 mg/l, less than 135.0 mg/l of Sodium and less than 180.0 mg/l of Sulfat

Irrigation: absence of Intestinal Helminth, a Faecal Coliform pollution that is below 30 mg/l, between 125.0 and 175.0 mg/l of Sodium and between 185.0 and 275.0 mg/l of Sulfat

These decisions are easy if we have access to the original database with its exact numerical values for the different parameters. The situation becomes difficult if we only have access to the discretized assessment values used for the generation of the colored map. In this case we cannot rely on a simple mapping from a combination of colors for the different toxin categories to possible uses, because the intended meaning of the colors that is needed for the decision is not accessible. However, if we manage to explicate the intended meaning of the colors, we have good chances of using the condensed information for decision making. In principle, the meaning of a color is encoded in the mapping tables shown above. To enable us to make use of this additional information, we have to provide comprehensive definitions of the concepts represented by the different colors. Using a logic-based representation these definitions could look as follows:

$$
\begin{aligned}
BacteriaGreen(W) &\iff IntestinalHelminth(W) =' no' \wedge \\
&\quad\; Faecal Coliforms(W) \leq 10.0 \\
BacteriaRed(W) &\iff IntestinalHelminth(W) =' yes' \vee \\
&\quad\; Faecal Coliforms(W) > 10.0 \\
\\
SaltsGreen(W) &\iff Sodium(W) \leq 200.0 \wedge \\
&\quad\; Sulfat(W) \leq 300.0 \\
SaltsRed(W) &\iff Sodium(W) > 200.0 \vee \\
&\quad\; Sulfat(W) > 300.0
\end{aligned}
$$

The above formulas define four categories a well W can belong to. These definitions can serve as input for logic reasoner to decide whether a well fulfills the requirement for a well that is suitable for one of the intended uses that have to be defined in the same way. Translating the informal requirements for the different kind of use into a formal definition that can be handled by a reasoner, we get:

$$
\begin{aligned}
Bathing(W) \quad &\Longleftrightarrow \quad IntestinalHelminth(W) =' no' \wedge \\
&\qquad FaecalColiforms(W) \leq 12.0
\end{aligned}
$$

$$
\begin{aligned}
Drinking(W) \quad &\Longleftrightarrow \quad IntestinalHelminth(W) =' no' \wedge \\
&\qquad FaecalColiforms(W) \leq 20.0 \wedge \\
&\qquad Sodium(W) \leq 135.0 \vee Sulfat(W) \leq 180.0
\end{aligned}
$$

$$
\begin{aligned}
Irrigation(W) \quad &\Longleftrightarrow \quad IntestinalHelminth(W) =' no' \wedge \\
&\qquad FaecalColiforms(W) \leq 30.0 \wedge \\
&\qquad Sodium(W) > 165.0 \wedge Sodium(W) \leq 200.0 \wedge \\
&\qquad Sulfat(W) > 245.0 \wedge Sulfat(W) \leq 300.0
\end{aligned}
$$

Using these definitions a logic reasoner is able to conclude that a well may be used for bathing, if the assessment value concerning the bacteria is 'green', because this means that Intestinal Helmith is absent and the level of Faecal Coliforms is below 10.0 and therefore also below 12.0. Concerning the use for drinking it can be concluded that drinking is not allowed if one of the assessments is 'red'. However, there is no definite result for the positive case, because we only know that Sodium is below 200.0 and Sulfate below 300.0 if both assessment values are 'green', while we demand them to be below 135.0 and 180, respectively. In practice, we would choose for a pessimistic strategy and conclude that drinking is not allowed, because of the risk of physical damage in the case of an incorrect result. The situation is similar for the irrigation case: we can decide that irrigation is not allowed if one of the assessment values is 'red'. Again, no definite result can be derived for the positive case. In this case it is likely that one would tend to an optimistic strategy, because the consequences of a failure is not as serious as they are for the drinking case.

2.5 The Role of Ontologies

In this section we want to propose ontologies as a technology for approaching the problem of explicating semantic knowledge about information. We first give a general overview of the nature and purpose of ontology that already reveals a great potential with respect to our task. Afterwards we sketch the idea of how ontologies could be used in order to support the semantic translation process. The idea presented will be elaborated in the remainder of the thesis.

The term 'Ontology' has been used in many ways and across different communities [Guarino and Giaretta, 1995]. If we want to motivate the use of ontologies for geographic information processing we have to make clear what we have in mind when we refer to ontologies. Thereby we mainly follow the description given in [Uschold and Gruninger, 1996]. In the following sections we

will introduce ontologies as an explication of some shared vocabulary or conceptualization of a specific subject matter. We will briefly describe the way an ontology explicates concepts and their properties and argue for the benefit of this explication in different typical application scenarios.

2.5.1 Shared Vocabularies and Conceptualizations

In general, each person has her individual view on the world and the things she has to deal with every day. However, there is a common basis of understanding in terms of the language we use to communicate with each other. Terms from natural language can therefore be assumed to be a shared vocabulary relying on a (mostly) common understanding of certain concepts with only little variety. This common understanding relies on the idea of how the world is organized. We often call this idea a 'conceptualization' of the world. Such conceptualization provide a terminology that can be used for communication.

The example of natural language already shows that a conceptualization is never universally valid, but rather for a limited number of persons committing to that conceptualization. This fact is reflected in the existence of different languages which differ more or less. For example, Dutch and German share may terms, however Dutch contains by far more terms for describing bodies of water, due to the great importance of water in the life of people. Things get even worse when we are not considered with every day language but with terminologies developed for special concerned areas. In these cases we often find situations where the same term refers to different phenomena. The use of the term 'ontology' in philosophy and its use in computer science may serve as an example. The consequence is a separation into different groups that share a terminology and its conceptualization. These groups are also called information communities [Kottmann, 1999] or ontology groups [Fensel et al., 1997]. An example of such a community is the $(KA)^2$ Initiative [Benjamins and Fensel, 1998].

The main problem with the use of a shared terminology according to a specific conceptualization of the world is that much information remains implicit. When a mathematician talks about the binomial $\binom{n}{k}$ he has much more in mind than just the formula itself. He will also think about its interpretation (the number of subsets of a certain size) and its potential uses (e.g. estimating the chance of winning in a lottery). Ontologies have set out to overcome the problem of implicit and hidden knowledge by making the conceptualization of a domain (e.g. mathematics) explicit. This corresponds to one of the definitions of the term ontology most popular in computer science [Gruber, 1993]:

"An ontology is an explicit specification of a conceptualization."

An ontology is used to make assumptions about the meaning of a term available. It can also be seen an an explication of the context a term is normally used in. Lenat [Lenat, 1998] for example describes context in terms of twelve independent

dimensions that have to be known in order to understand a piece of knowledge completely and shows how these dimensions can be explicated using the Cyc ontology.

2.5.2 Specification of Context Knowledge

There are many different ways in which an ontology may explicate a conceptualization and the corresponding context knowledge. The possibilities range from a purely informal natural language description of a term corresponding to a glossary up to strictly formal approaches with the expressive power of full first order predicate logic or even beyond (e.g. Ontolingua [Gruber, 1991]). Jasper and Uschold distinguish two ways in which the mechanisms for the specification of context knowledge by an ontology can be compared [Jasper and Uschold, 1999]:

Level of Formality The specification of a conceptualization and its implicit context knowledge can be done at different levels of formality. As already mentioned above, a glossary of terms can also be seen as an ontology despite its purely informal character. A first step to gain more formality is to prescribe a structure to be used for the description. A good example for this approach is the new standard web annotation language XML [Bray et al., 1998]. XML offers the possibility to define terms and organize them in a simple hierarchy according to the expected structure of the web document to be described in XML. The organization of the terms is called a Data Type Definitions DTD. However, the rather informal character of XML encourages its misuse. While the hierarchy of an XML specification was originally designed to describe layout it can also be exploited to represent sub-type hierarchies [van Harmelen and Fensel, 1999] which may lead to confusion. This problem can be solved by assigning formal semantics to the structures used for the description of the ontology. An example is the conceptual modeling language CML [Schreiber et al., 1994]. CML offers primitives to describe a domain that can be given a formal semantics in terms of first order logic [Aben, 1993]. However a formalization is only available for the structural part of a specification. Assertions about terms and the description of dynamic knowledge is not formalized, offering total freedom for the description. On the other extreme there are also specification languages which are completely formal. A prominent example is Ontolingua (see above), one of the first Ontology languages which is based on the Knowledge Interchange Format KIF [Genesereth and Fikes, 1992] which was designed to enable different knowledge-based systems to exchange knowledge.

Extend of Explication The other comparison criterion is the extend of explication that is reached by the ontology. This criterion is strongly connected with the expressive power of the specification language used. We already mentioned DTD's which are mainly a simple hierarchy of terms. We can generalize this by saying that the least expressive specification of an ontology consists of an organization of terms in a network using two-placed

relations. This idea goes back to the use of semantic networks. Many extensions of the basic idea have been proposed. One of the most influential was the use of roles that could be filled out by entities showing a certain type [Brachman, 1977]. This kind of value restriction can still be found in recent approaches. RDF schema descriptions [Champin, 2000] which might become a new standard for the semantic descriptions of web-pages is an example. An RDF schema contains class definitions with associated properties that can be restricted by so-called constraint-properties. However, default values and value range descriptions are not expressive enough to cover all possible conceptualizations. A greater expressive power can be provided by allowing classes to be specified by logical formulas. These formulas can be restricted to a decidable subset of first order logic. This is the approach of so-called description logics [Donini et al., 1996]. Nevertheless, there are also approaches allowing for more expressive descriptions. In Ontolingua for example, classes can be defined by arbitrary KIF-expressions. Beyond the expressiveness of full first-order predicate logic there are also special purpose languages that have an extended expressiveness to cover specific needs of their application area.

2.5.3 Beneficial Applications

Ontologies are useful for many different applications that can be classified into several areas [Jasper and Uschold, 1999]. Each of these areas has different requirements on the level of formality and the extend of explication provided by the ontology. The common idea of all of these applications is to use ontologies in order to reach a common understanding of a particular domain. In contrast to syntactic standards the understanding is not restricted to a common representation or a common structure. The use of ontologies also help to reach a common understanding of the *meaning* of terms. Therefore ontologies are a promising candidate in order to support semantic interoperability. We will shortly review some common application areas namely the support of communication processes, the specification of systems and information entities and the interoperability of computer systems.

Communication Information communities are useful, because they ease communication and cooperation among its members by the use of a shared terminology with a well defined meaning. On the other hand, the formation of information communities makes communication between members from different information communities very difficult, because they do not agree on a common conceptualization. They may use the shared vocabulary of natural language. However most of the vocabulary used in their information communities is highly specialized and not shared with other communities. This situation demands for an explication and explanation of the terminology used. Informal ontologies with a large extend of explication are a good choice to overcome these problems. While definitions have always played an important role in scientific literature, conceptual models of certain domains are rather

new. However nowadays systems analysis and related fields like software engineering rely on conceptual modeling to communicate structure and details of a problem domain as well as the proposed solution between domain experts and engineers. Prominent examples of ontologies used for communication are Entity-Relationship diagrams [Chen, 1976] and Object-oriented Modeling languages like UML [Rumbaugh et al., 1998].

Systems Engineering Entity Relationship diagrams as well as UML are not only used for communication, they also serve as building plans for data and systems guiding the process of building (engineering) the system. The use of ontologies for the description of information and systems has many benefits. The ontology can be used to identify requirements as well as inconsistencies in a chosen design. It can help to acquire or search for available information. Once a systems component has been implemented its specification can be used for maintenance and extension purposes. Another very challenging application of ontology-based specification is the reuse of existing software. In this case the specifying ontology serves as a basis to decide if an existing component matches the requirements of a given task [Motta, 1999]. Depending on the purpose of the specification, ontologies of different formal strength and expressiveness are to be used. While the process of communicating design decisions and the acquisition of additional information normally benefit from rather informal and expressive ontology representations (often graphical), the directed search for information needs a rather strict specification with a limited vocabulary to limit the computational effort. At the moment, the support of semi- automatic software reuse seems to be one of the most challenging applications of ontologies, because it requires expressive ontologies with a high level of formal strength (see for example [van Heijst et al., 1997]).

Interoperability The above considerations might provoke the impression that the benefits of ontologies are limited to systems analysis and design. However, an important application area of ontologies is the integration of existing systems. The ability to exchange information at run time, also known as interoperability, is an important topic. The attempt to provide interoperability suffers from problems similar to those associated with the communication amongst different information communities. The important difference is that the actors are not persons able to perform abstraction and common sense reasoning about the meaning of terms, but machines. In order to enable machines to understand each other we also have to explicate the context of each system, but on a much higher level of formality in order to make it machine understandable (The KIF language was originally defined for the purpose of exchanging knowledge models between different knowledge based systems). Ontologies are often used as inter-linguas for providing interoperability [Uschold and Gruninger, 1996]: They serve as a common format for data interchange. Each system that wants to inter-operate with other systems has to transfer its information into this common framework.

Information Retrieval Common information-retrieval techniques either rely on a specific encoding of available information (e.g. fixed classification codes) or simple full-text analysis. Both approaches suffer from severe shortcomings. First of all, both completely rely on the input vocabulary of the user which might not be completely consistent with the vocabulary of the information. Second, a specific encoding significantly reduces the recall of a query, because related information with a slightly different encoding is not matched. Full-text analysis on the other hand reduces precision, because the meaning of the words might be ambiguous.

Using an ontology in order to explicate the vocabulary can help overcome some of these problems. When used for the description of available information as well as for query formulation an ontology serves as a common basis for matching queries against potential results on a semantic level. The use of rather informal ontologies like WordNet [Fellbaum, 1998] increases the recall of a query by including synonyms into the search process. The use of more formal representations like conceptual graphs [Sowa, 1999] further enhances the retrieval process, because a formal representation can be used to increase recall by reasoning about inheritance relationships and precision by matching structures. To summarize, information retrieval benefits from the use of ontologies. Ontologies help to de-couple description and query vocabulary and increases precision as well as recall [Guarino et al., 1999].

Conclusion

Interoperability between different information sources is an important topic with regard to the efficient sharing and use of information across different systems and applications. While many syntactic and structural problems of the integration process that is inevitable for achieving interoperability have been solved the notion of *semantic interoperability* still bears serious problems. Problems on the semantic level occur due to the inherent context dependency of information that can only be understood in the context of their original source and purpose. The main problem with context dependencies with respect to semantic interoperability is the fact that most of the contextual knowledge that is necessary for understanding the information is hidden in documentation and specification of an information source: it remains implicit from the view of the actual information. The only way to overcome this problem is the use of an explicit context model that can be used to re-interpret information in the context of a new information source and a new application. The use of ontologies is a straightforward and promising approach in order to explicate contextual information and to make a semantics preserving translation possible. In especially, ontologies could be used for the specification of a source independent shared vocabulary (domain ontology) whose concepts are used to describe the specific contextual information of different information sources to be integrated (application ontologies). The use of a common vocabulary as a basis for the

context specifications is assumed to enable us to to perform (semi-) automatic translations between different contexts that preserve the intended meaning of the translated terms to a large extend.

Part I

Language Interoperability

Chapter 3

Ontology Languages

In this chapter we lay the foundation for answering the question of how ontologies should be represented and how reasoning can be performed in order to support information sharing. We present ontology languages that are compatible with existing Web technology and compare them to a wider range of existing language. We conclude that despite all standardization efforts, there are still significant differences with respect to expressiveness and reasoning capabilities.

Acknowledgement: The descriptions of RDF schema and DAML+OIL in the sections 3.1 and 3.2 have been adopted from a tutorial given at K-CAP'01 together with Dieter Fensel and Frank van Harmelen.

Using explicit models of information semantics for information sharing requires means for encoding these models. Figure 3.1 illustrates this encoding: information from a document is translated into formal logic using a controlled language that defines logical operators and terms that can be used to describe information items and helps to map their semantics into a formal logic. On the other side a controlled language is used to perform the transition from the formal logic back to information in a different document.

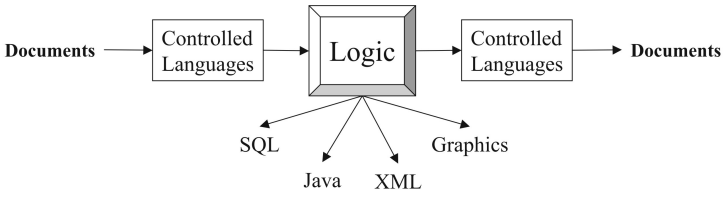


Figure 3.1: Exchange of Information [Sowa, 2000]

The need for using controlled languages as a means for mapping metadata

models on logic has also been recognized by the W3C. Their proposal for a an advanced Web architecture includes two controlled languages on top of XML and RDF which are used to describe data and metadata, respectively.

The first language mentioned is RDF schema [Brickley et al., 1998], a set of predefined RDF properties for expressing terminological knowledge. On top of RDF schema DAML+OIL [van Harmelen et al., 2001] has been defined as a rich language for expressing ontological vocabularies. The language is designed in such a way that it can easily be mapped onto a formal logic which constitutes the next layer of the architecture.

3.1 RDF Schema

As described in the introduction, RDF is build upon the notion of resources, information units that can have certain properties with corresponding values. In turn, modeling elements of the RDF language are also resources. RDF schema refines the notion of modeling resources. In especially, RDF schema defines standard properties, constraint properties and classes. Figure 3.2 gives and overview of the introduced resources that will be discussed in the following.

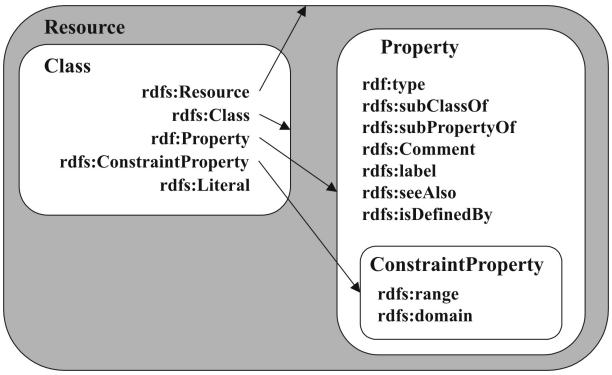


Figure 3.2: Modeling Components of RDF schema from [Brickley et al., 1998]

In the following, we use the domain of family relations adapted from [Patel-Schneider and Swartout, 1994], [Haarslev and Moller, 2001] and [van Harmelen et al., 2001] in order to illustrate the modeling constructs of the languages we discuss.

3.1.1 Relations

The main descriptive element of RDF are properties of resources specified by the RDF resource `rdf:property` specified in the RDF name space. Properties are

used to describe arbitrary binary relations between resources. In our domain, we could for example describe the following family relations using RDF properties:

```
<rdf:Description rdf:about="alice">
    <has-child rdf:resource="#betty>
    <has-child rdf:resource="#charles>
</rdf:Description>

<rdf:Description rdf:about="betty">
    <has-child rdf:resource="#doris>
    <has-child rdf:resource="#eve>
</rdf:Description>

<rdf:Description rdf:about="charles">
    <has-sibling rdf:resource="#betty>
</rdf:Description>

<rdf:Description rdf:about="doris">
    <has-sister rdf:resource="#eve>
</rdf:Description>

<rdf:Description rdf:about="eve">
    <has-sister rdf:resource="#doris>
</rdf:Description>
```

Using these properties, we can describe members of a family as resources and relate them by the usual family relation like `has-child`, `has-sibling` and `has-sister`. RDF schema now defines the special property `rdfs:subPropertyOf` that can be used to define a specialization of an existing property. In our example, we can state that `has-sister` and `has-brother` are special cases of `has-sibling` and that `has-child` is a special case of `has-descendant` in the following way:

```
<rdf:property id="has-sister">
    <rdfs:subPropertyOf rdf:resource="#has-sibling"/>
</rdf:property>

<rdf:property id="has-brother">
    <rdfs:subPropertyOf rdf:resource="#has-sibling"/>
</rdf:property>

<rdf:property id="has-child">
    <rdfs:subPropertyOf rdf:resource="#has-descendant"/>
</rdf:property>
```

The intended meaning of this new modeling construct can be used to derive new knowledge from an existing model. In our example case we can add the following facts to our family model:

```
<rdf:Description rdf:about="alice">
    <has-descendant rdf:resource="#betty/>
    <has-descendant rdf:resource="#charles/>
</rdf:Description>

<rdf:Description rdf:about="betty">
    <has-descendant rdf:resource="#doris/>
    <has-descendant rdf:resource="#eve/>
</rdf:Description>

<rdf:Description rdf:about="doris">
    <has-sibling rdf:resource="#eve/>
</rdf:Description>

<rdf:Description rdf:about="eve">
    <has-sibling rdf:resource="#doris/>
</rdf:Description>
```

This ability to infer new knowledge in a standardized way is already a big advantage over the use of plain RDF, where all conclusions to be made from a model had to be specified individually for different systems.

3.1.2 Type Restrictions

The RDF language already contains a possibility to assign type information to resources using the `rdf:type` property. In plain RDF models type information has to be explicitly assigned to resources using this property. Beyond this, RDF schema allows us to connect type information with properties (binary relations). Any property can be restricted by an `rdfs:range` and an `rdfs:domain` statement. The first one forces the second resource in the relation to belong to a certain concept, the latter enforces a certain type for the first resource in the relation. In our example, we might want to add the information that people with children belong to the class of parents as well as that brothers are male and sisters are female persons:

```
<rdf:Description about="#has-child">
    <rdfs:domain rdf:resource="#parent"/>
</rdf:Description>

<rdf:Description about="#has-brother">
    <rdfs:range rdf:resource="#male-person"/>
</rdf:Description>

<rdf:Description about="#has-sister">
    <rdfs:domain rdf:resource="#female-person"/>
</rdf:Description>
```

These restrictions can be used to derive some type information for the resources in our family model:

```
<rdf:Description rdf:about="alice">
    <rdf:type rdf:resource="#parent"/>
</rdf:Description>

<rdf:Description rdf:about="betty">
    <rdf:type rdf:resource="#parent"/>
</rdf:Description>

<rdf:Description rdf:about="doris">
    <rdf:type rdf:resource="#female-person"/>
</rdf:Description>

<rdf:Description rdf:about="eve">
    <rdf:type rdf:resource="#female-person"/>
</rdf:Description>
```

We can see that the extended capabilities for specifying relation with RDF schema already provides means to encode simple terminological knowledge. However, the language so far only helps to assign resources to certain types, what is still missing are means to impose additional structure on classes.

3.1.3 Class Hierarchies

While the RDF language contains the `rdf:type` operator, there is no explicit notion of classes. RDF schema fills this gap by introducing classes as special kinds of resources. They are identified by the resource `rdfs:Class`. A general resource can be identified as a class using the `rdf:type` property. RDF schema also defines the `rdfs:subClassOf` property for specifying taxonomic hierarchies. In our example, we could define a hierarchy of different kinds of people starting with the general concept person and defining parents as well as male persons and female persons to be subclasses of this general concept. RDF schema allow multiple inheritance. We thus can define a mother to be a subclass of parent as well as female person:

```
<rdfs:Class rdf:ID="parent">
    <rdfs:subClassOf rdf:resource="#person"/>
</rdfs:Class>

<rdfs:Class rdf:ID="male-person">
    <rdfs:subClassOf rdf:resource="#person"/>
</rdfs:Class>

<rdfs:Class rdf:ID="female-person">
    <rdfs:subClassOf rdf:resource="#person"/>
```

```
</rdfs:Class>

<rdfs:Class rdf:ID="mother">
    <rdfs:subClassOf rdf:resource="#parent"/>
    <rdfs:subClassOf rdf:resource="#female-person"/>
</rdfs:Class>
```

Using these additional statements we can infer that all of the resource in our example model belong to the class person:

```
<rdf:Description rdf:about="alice">
    <rdf:type rdf:resource="#person"/>
</rdf:Description>

<rdf:Description rdf:about="betty">
    <rdf:type rdf:resource="#person"/>
</rdf:Description>

<rdf:Description rdf:about="doris">
    <rdf:type rdf:resource="#person"/>
</rdf:Description>

<rdf:Description rdf:about="eve">
    <rdf:type rdf:resource="#person"/>
</rdf:Description>
```

Summarizing, RDF schema provide a controlled vocabulary for specifying the terminological structure of an application domain with an axiomatic semantics that can be implemented in a formal logic in order to provide simple inference services like type checking or taxonomic reasoning.

3.2 DAML+OIL

The example domain already shows that there are many aspects of terminological knowledge that can not be captured by RDF schema. These aspects include the following fact:

- the same person may not be male and female
- every person has exactly one mother
- a female person automatically becomes a mother when having a child
- has-child is the inverse relation of has parent

Including these facts in a model of information semantics requires a far more expressive language. DAML+OIL [van Harmelen et al., 2001] is such a language

that has been defined on top of RDF schema, extending it with additional operators for defining class, relations and individuals. A DAML+OIL model uses RDF schema elements whenever possible, additionally it defines a namespace that contains other operators we will discuss in the following paragraphs.

3.2.1 Class Building Operations

The only possibility to define class structures in RDF schema was the `rdfs:subClassOf` property. DAML+OIL adopts this relation also allowing for multiple inheritance and provides a property for stating that two classes are disjoint. Using this additional property, we can refine the definition of male and female persons:

```
<daml:Class rdf:ID="Male">
    <rdfs:subClassOf rdf:resource="#Animal"/>
</daml:Class>

<daml:Class rdf:ID="Female">
    <rdfs:subClassOf rdf:resource="#Animal"/>
    <daml:disjointWith rdf:resource="#Male"/>
</daml:Class>

<daml:Class rdf:ID="Man">
    <rdfs:subClassOf rdf:resource="#Person"/>
    <rdfs:subClassOf rdf:resource="#Male"/>
</daml:Class>
```

The expressiveness of the subclass relation in DAML+OIL is further enriched be the possibility of defining a class to be equivalent to a logical expression over class names. As an example, we could define the class of persons to be the disjoint union of the members of the classes of men and women.

```
<daml:Class rdf:about="#Person">
    <daml:disjointUnionOf rdf:parseType="daml:collection">
        <daml:Class rdf:about="#Man"/>
        <daml:Class rdf:about="#Woman"/>
    </daml:disjointUnionOf>
</daml:Class>
```

Beside the `daml:disjointUnionOf` property, classes can also be defined to be equivalent to another class, to equivalent to a Boolean expression over classes using `daml:intersectionOf`, `daml:unionOf` and `daml:complementOf` or by enumerating its elements with the `daml:oneOf` property.

3.2.2 Relations

DAML+OIL defines two kinds of relations. `daml:ObjectProperty` relates members of different classes to each other. `daml:DatatypeProperty` relates a mem-

ber of a class to a legal value of a certain data type. The first type of relation is very similar to an RDF property. It has a unique name and can have RDF schema range and domain restrictions like the following example:

```
<daml:ObjectProperty rdf:ID="hasParent">
    <rdfs:domain rdf:resource="#Animal"/>
    <rdfs:range rdf:resource="#Animal"/>
</daml:ObjectProperty>
```

The first enhancement to RDF schema employed by DAML+OIL is the possibility of defining one relation to be the equivalent or the inverse of another relation. Using this feature, we can define the `has-child` relation using the one specified above:

```
<daml:ObjectProperty rdf:ID="hasChild">
    <daml:inverseOf rdf:resource="#hasParent"/>
</daml:ObjectProperty>
```

Just as RDF schema, hierarchies of relations can be specified using the `rdfs:subpropertyOf` operator. Further, special properties can be assigned to relations which may be sub-relations of other relations that don't need to have that special property, like in the example below.

```
<daml:UniqueProperty rdf:ID="hasMother">
    <rdfs:subPropertyOf rdf:resource="#hasParent"/>
    <rdfs:range rdf:resource="#Female"/>
</daml:UniqueProperty>
```

In this case, the `hasMother` relation is defined to be a `daml:UniqueProperty` which means that there can only be one mother of someone. In turn, `daml:UnambigousProperty` would state the same fact for the inverse property. The last special property is the transitivity of a relation defined by `daml:transitiveProperty` which would for example apply to the `has-descendant` property mentioned in the last section.

3.2.3 Property Restrictions

Classes define common properties of its members. Different from RDF schema, DAML+OIL provides means for defining these characteristic properties of class members in terms of restrictions on the objects they are related to. In principle there are two kinds of restrictions, type restrictions and number restrictions. Using these restrictions, we can for example define that the father of any person is a man and that there is only one father for any person:

```
<daml:Class rdf:ID="Person">
    <rdfs:subClassOf rdf:resource="#Animal"/>
    <rdfs:subClassOf>
```

```
            <daml:Restriction>
                <daml:onProperty rdf:resource="#hasFather"/>
                <daml:toClass rdf:resource=#Man/>
            </daml:Restriction>
            <daml:Restriction daml:cardinality="1">
                <daml:onProperty rdf:resource="#hasFather"/>
            </daml:Restriction>
        </rdfs:subClassOf>
    </daml:Class>
```

The restriction `daml:toClass` from the example claims that every object related to a member of the class has to be of a certain type. Beside this restriction, `daml:hasClass` claims that every member of the class is related to at least one object of a certain type, `daml:hasValue` even claims that every object of the class is related to one specific object. Number restrictions `daml:minCardinality`, `daml:maxCardinality` and `daml:cardinality` define lower and upper boundaries and exact values for the number of objects the member of a class is related to via a certain relation. Several restrictions may apply to a relation. In order to reduce the size of specifications, DAML+OIL defines the restrictions `daml:minCardinalityQ`, `daml:maxCardinalityQ` and `daml:cardinalityQ` that combine type and number restrictions. Using these restrictions the class defined above can be specified as follows:

```
<daml:Class rdf:ID="Person">
    <rdfs:subClassOf rdf:resource="#Animal"/>
    <rdfs:subClassOf>
        <daml:Restriction daml:cardinalityQ="1">
            <daml:onProperty rdf:resource="#hasFather"/>
            <daml:hasClassQ rdf:resource=#Man/>
        </daml:Restriction>
    </rdfs:subClassOf>
</daml:Class>
```

Type restrictions do not have to refer to a class name, they can use the full expressive power of DAML+OIL in order to describe the class a related object has to belong to. This class in turn is described by property restrictions.

3.2.4 Data Types

The specification of some properties of objects such as age or size of a person requires standard data types such as integers or real numbers. For this purpose DAML+OIL uses the data type specification capabilities of XML schema. Using XML schema, we can for example define a special data type containing all natural numbers greater or equal to the number eighteen.

```
<xsd:simpleType name="over17">
    <xsd:restriction base="xsd:positiveInteger">
```

```
        <xsd:minInclusive value="18"/>
    </xsd:restriction>
</xsd:simpleType>
```

These kind of user defined data types can be used to define common properties of class members. Using the data type `over17` in order to define the class of adult people by restricting the value of the age relation to be from that data type and therefore to be over seventeen:

```
<daml:Class rdf:ID="Adult">
    <daml:intersectionOf rdf:parseType="daml:collection">
        <daml:Class rdf:about="#Person"/>
        <daml:Restriction>
            <daml:onProperty rdf:resource="#age"/>
            <daml:hasClass rdf:resource=...#over17"/>
        </daml:Restriction>
    </daml:intersectionOf>
</daml:Class>
```

As already mentioned above, the corresponding property has to be defined to be a data type property, because it is normally handled different from object properties by reasoning methods.

3.2.5 Individuals

Beside the definition of classes and relations DAML+OIL is capable of describing individuals using an RDF notation and of specifying values for their object and data type properties. Therefore the following RDF statement is a legal expression in DAML+OIL:

```
<father rdf:ID="Hans Meier">
    <has-brother rdf:resource="#Udo Meier"/>
    <age><xsd:integer rdf:value="42"/></age>
</father>
```

DAML+OIL extends the basic RDF vocabulary for describing resources by the `daml:differentIndividualFrom` property for explicitly stating that two individuals are not the same. Using this option we can define the brother of Hans Meier to be a different individual and not himself:

```
<man rdf:ID="Udo Meier">
    <daml:differnetIndividualFrom rdf:resource="#Hans Meier"/>
</man>
```

A further extension made by DAML+OIL is the possibility to state that two individuals are the same even if they do not have the same ID:

```
<rdf:Description rdf:ID="Uncle Hans">
    <daml:sameIndividual rdf:resource="#Hans Meier"/>
</rdf:Description>
```

The specification above introduces the new individual called 'Uncle Hans' that is in fact only a different name for the person previously introduced as Hans Meier.

3.3 Other Web-Based Ontology Languages

Besides the two standards discussed above a number of other approaches for encode ontologies on the World Wide Web have been proposed. In a early article van Harmelen and Fensel review the ability of conventional Web technology for knowledge representation on the Web [van Harmelen and Fensel, 1999]. They conclude that the abilities of these technologies are rather restricted. Following this observation some specialized languages have been proposed to overcome this shortcoming. A comparison of these languages, including RDF schema and DAML+OIL is reported in [Gomez-Perez and Corcho, 2002]. We will now briefly review the results of this comparison and discuss implications for our work.

3.3.1 Semantic Web Languages

Besides RDF schema and DAML+OIL, which have been introduced above, the comparison reported in [Gomez-Perez and Corcho, 2002] includes the following languages that have been selected on the basis of their aim of supporting knowledge representation on the Web and their compatibility to the Web standards XML or RDF.

XOL: (XML-based Ontology Language) [Karp et al., 2002] has been proposed as a language for exchanging formal knowledge models in the domain of bio-informatics. The development of XOL has been guided by the representational needs of the domain and by existing frame-based knowledge representation languages.

SHOE: (Simple HTML Ontology Extension) [Luke and Hefflin, 2002] was created as an extension of HTML for the purpose of defining machine-readable semantic knowledge. The aim of SHOE is to enable intelligent Web agents to retrieve and gather knowledge more precisely than it is possible in the presence of plain HTML documents.

OML: (Ontology Markup Language) [Kent, 2002] is an ontology language that has initially been developed as an XML serialization of SHOE. Meanwhile, the language consists of different layers with increasing expressiveness. The semantics especially of the higher levels is largely based on the notion of conceptual graphs. In the comparison, however, only a less expressive subset of OML (simple OML) is considered.

OIL: (Ontology Inference Layer) [Fensel et al., 2001] is an attempt to develop an ontology language for the Web that has a well defined semantics and sophisticated reasoning support for ontology development and use. The language is constructed in a layered way starting with core-OIL, providing a formal semantics for RDF schema, standard-OIL which is equivalent to an expressive description logic with reasoning support and Instance OIL that adds the possibility of defining instances.

We have to mention that there is a strong relationship between the OIL language and RDF schema as well as DAML+OIL. OIL extends RDF schema and has been the main influence in the development if DAML+OIL. The main difference between OIL and DAML+OIL is an extended expressiveness of DAML+OIL in terms of complex definitions of individuals and data types.

3.3.2 Comparison and Results

The comparison of the languages mentioned above was carried out on the basis of the set of elements contained in the language and their ability to encode semantic information about a domain in terms of the following aspects:

Concepts and Taxonomies: Ontologies usually group objects of the world that have certain properties in common (e.g. cities or countries). A description of the shared properties is called a concept definition. Concepts can be arranged into a subclass-superclass relation in order to be able to further discriminate objects into sub-groups (e.g. capitals or European countries).

Relations: Objects of the same type normally occur in similar situations where they have a certain relation to each other (cities lie in countries, countries have a capital). These typical relations can often be specified in order to establish structures between groups of objects.

Instances: Sometimes single objects (e.g. the continent Europe) play a prominent role in a domain of interest or the membership to a concept is defined by the relation to a specific object (European countries are those contained in Europe). For this purpose ontology languages often allow to specify single objects, also called instances.

Axioms: Sometimes a domain follows certain rules that cannot be expressed with the elements discussed above (e.g. the fact that the number of inhabitants of Europe equals the sum of the number of inhabitants of European countries). In order to capture these rules some languages allow to specify axioms in a formal logic.

The comparison revealed significant differences in terms of expressiveness of the different languages. An overview of the results with respect to class definitions and taxonomies is given in figure 3.3. The authors suggest that before choosing one of the languages, an analysis of th representational needs

of a particular application has to be carried out. The result of this analysis should guide the selection of one of the languages considered in the comparison. Assuming that developers of Web-based information follow this suggestion, we will have to be able to handle all of these languages in order to support information sharing.

	XOL	**SHOE**	**OML**	**RDS/S**	**OIL**	**DAML+OIL**
Partitions	-	-	+	-	+	+
Attributes						
Instance Attr.	+	+	+	+	+	+
Class Attr.	+	-	+	-	+	+
Local Scope	+	+	+	+	+	+
Global Scope	+	-	+	+	+	+
Facets						
Default Values	+	-	-	-	-	-
Type Constr.	+	+	+	+	+	+
Cardinalities	+	-	-	-	+	+
Taxonomies						
Subclass of	+	+	+	+	+	+
Exhaustive Comp.	-	-	+	-	+	+
Disjoint Comp.	-	-	+	-	+	+
Not Subclass of	-	-	-	-	+	+

Figure 3.3: Comparison of Web Ontology Languages with respect to Concepts and Taxonomies (taken from [Gomez-Perez and Corcho, 2002])

This observation appears to be quite discouraging. However, the comparison also reveals that DAML+OIL is the most expressive language for encoding ontologies on the Web. Therefore we can assume that if we are able to handle DAML+OIL models, we should in principle also be able to understand and process models in the other languages, provided that we can establish a formal framework for comparing and relating ontology languages.

Conclusions

We reviewed existing languages for encoding ontologies on the World Wide Web, focusing on the two proposed standards RDF schema and DAML+OIL. These languages are especially suited for our purposes, because they are well integrated with syntactic standards of the World Wide Web, i.e. XML and RDF. The existence of other languages such as SHOE, XOL or OML implies that there is a need for language integration if we want to fulfill the promise of being able to facilitate information sharing across existing information repositories, because they may use any of the languages mentioned. A comparison of the existing languages revealed that DAML+OIL is the most expressive language

that is currently used on the Web. From this observation we conclude that an approach for language interoperability has to be able to handle at least the expressiveness of DAML+OIL.

Chapter 4

Language Interoperability

In the previous chapter we discussed languages for encoding information semantics using explicit models of terminological knowledge. We concluded that there is a need for integrating languages if we want to facilitate information sharing. In this chapter, we define a general framework for ensuring interoperability between ontology languages. The goal is to show that we can translate between different languages without loss of information, thus providing a formal basis for language interoperability. Further, we briefly describe how the framework can be implemented using existing Web technologies which is a requirement for the applicability with respect to information sharing on the Web.

Acknowledgements: Parts of this chapter have been published before. The idea of integrating ontology language based on transformations that preserve certain formal properties has been discussed in [Stuckenschmidt and Euzenat, 2001]. A discussion of the different formal properties of transformations can be found in [Euzenat and Stuckenschmidt, 2003].

When system designers choose a language for encoding information semantics, their choice is driven by various decisions. Different types of knowledge can be used for different kinds of reasoning tasks. Further different kinds of reasoning methods result in different levels of reasoning complexity. In order to choose a language, the designer has to take design constraints into account that originate from the application at hand. Examples for design constraints are:

- The conceptualization of the application domain as well as pre-existing models imply the existence of certain knowledge types. The designed language must at least implicitly support these knowledge types.
- The role of the ontology in the overall application implies a certain task type. The design space is therefore restricted to variations of this task type.

- The availability of reasoners for the given task does not only have impact on the reasoning complexity, but also on the types of knowledge that can be used to define the ontology.

In order to support information sharing across heterogeneous systems, the different choices made by system designers force us to handle different languages that can be categorized as follows:

System Languages: The different systems we want to enable to exchange their information may use different ontology languages due to diverging representational needs, design decisions or organizational conventions.

Reasoner Languages: In order to support information sharing, we might need to reason about ontologies. Available inference engines for this purpose are normally specialized to handle a specific language that is not necessarily the same that one of the systems uses.

Integration Languages: Often, intermediate languages are used to integrated models encoded in different languages that are not directly translatable into each other without problems. We might have to include or generate such intermediate languages in the course of integration.

In this chapter, we describe a general framework for integrating different ontology language by defining transformations between elements of these languages. In section 4.1 we first briefly review existing architectures for the language integration problem from an architectural point of view and propose the so-called family of languages approach as a generalization of these approaches that leaves more flexibility for the integration process. We then review a common formal basis for ontology languages that has been proposed in [Baader et al., 2000] and show that the framework is general enough to capture the DAML+OIL language. Equipped with the common formal basis we investigate transformations between ontology languages and their formal properties. Finally, we show how transformations between languages can be implemented using technology described in [Euzenat, 2001].

4.1 The 'Family of Languages' Approach

In this section, we introduce the family of languages approach to semantic interoperability. This approach builds upon existing architecture and generalizes them, thus providing greater flexibility for language integration. In the following, we formally characterize the different architectures. Then, we introduce our approach on an abstract level and show that it generalizes the other characterizations. This general description will be the basis for the concrete transformation approach for terminological languages described in the next section.

4.1.1 Transformation Architectures

We consider the problem of integrating a set $\{L_1, \cdots, L_n\}$ of different languages using arbitrary transformations $\tau : L_{i_1} \rightarrow L_{i_2}$. Further, we assume that the goal is to find a language L' to be used in an application and to find transformation from every language $L_1, \cdots, L_n$ into L' in order to be able to transform knowledge encoded in these languages into a common format. We refer to this ability in terms of a coverage relation between languages:

Definition 4.1 (Language Coverage) *A language L' is said to cover another language L if there exists a mapping $\tau : L \rightarrow L'$ such that*

$$\exists \tau, (\forall \delta \in L, \tau(\delta) \in L') \tag{4.1}$$

We denote this fact by $L \prec L'$.

We found that different approaches to knowledge sharing differ in the way the language L' and the transformations are chosen.

The Mapping Approach The most direct and often used approach matches certain types of expressions in the source language and creates corresponding expressions in the target language. The formal nature of these mappings vary from purely syntactic matches to so-called theory interpretations with well defined properties. Therefore, we characterize the mapping approach solely by the existence of a function that maps expressions from one language to another.

This approach has the drawback of requiring transformations from any language to any other. It is thus not very reusable and require to check the properties of the transformation individually. A current example of the mapping approach is the OntoMorph system described in [Chalupsky, 2000].

The Pivot Approach In order to reduce the number of transformations necessary to integrate a certain number of languages, a special transformation architecture can be used. One of the most common is the use of a single pivot language L_P all other languages are translated to. In order to be able to preserve semantics, this pivot language has to cover all other languages. More formally, the pivot approach is characterized by the following assumption:

$$\exists! L_P, \forall L_i, (L_i \prec L_P) \tag{4.2}$$

Probably the most prominent example of a pivot architecture is Ontolingua [Gruber, 1993]. In this approach the Ontolingua language serves as a pivot language. However, translations are also performed from Ontolingua into less expressive languages leading to a loss of information the approach has often been criticized for.

The Layered Approach A third approach to deal with semantic interoperability is the use of a layered architecture containing languages with increasing expressiveness. This approach has been proposed in order to avoid the problems arising from the need of using a very expressive language and to ensure tractable reasoning with the integrated languages. In such a layered architecture, representations can be translated into languages higher in the hierarchy without loss of information. Formally speaking, the languages form a total order induced by the coverage relation.

$$\forall i, j, (i \leq j \Rightarrow L_i \prec L_j) \tag{4.3}$$

A recent example of a layered architecture is the ontology language OIL [Fensel et al., 2000] that has been built onto existing Web standards. The idea is to use the W3C standard RDF schema as the language on the lowest layer and build additional language features on top of it. Doing this, it is possible to translate RDF schema definitions into languages of the higher levels in order to enrich them.

4.1.2 A Generalization

The idea behind the family of languages approach presented in this section is to have a set of languages structured by a partial order. This is more general than the layering approach and more convenient for the users who can find languages closer to their needs (or, for an intermediate language, languages closer to their own languages). In order to fulfill the translation task, for every two languages in the family a third language should exist that covers both of them. This third language is the one both languages can be translated into thereby achieving interoperability.

Definition 4.2 (Family of Languages Property) *A set of languages $L_1, \cdots, L_m$ satisfies the family of language property, iff they form a semi-lattice with respect to the coverage relation, i.e. if for every pair of languages $L_1, L_2 \in \{L_1, \cdots, L_m\}$ there exists a language $L \in \{L_1, \cdots, L_m\}$ such that*

$$(L_1 \prec L) \wedge (L_2 \prec L) \tag{4.4}$$

In fact, the family of languages approach is a generalization of the pivot and the layered approach that further increases the flexibility of the transformation process. The approach generalizes the pivot approach insofar as the pivot approach fulfills the family of languages property, because the pivot language L_P can always be used as integration language. It also generalizes the layered approach, because in the layered framework the language that is higher in the hierarchy can be used as the integration language in the sense of the family of languages property. However, the family of languages approach is more flexible, because it does not require a fixed pivot language nor a fixed layering

of language. On the contrary, any language that fulfills certain formal criteria can be used as integration language. We discuss these formal criteria in the following section.

Proposition 1 (Generality of the Approach) *The family of languages property generalizes the pivot and the layered approach to language integration. In particular, it is implied by equation 4.2 as well as 4.3.*

Proof 1 *Without loosing generality, we choose two languages L_i and L_j to be integrated. We have to find a language L' with $L_i \prec L'$ and $L_j \prec L'$.*

(4.2 $\Longrightarrow$ 4.4) *From equation 4.2 we know that there exists a language L_p with $L_1 \prec L_p$ and $L_2 \prec L_p$. Therefore the family of languages property holds for any pivot approach for $L' = L_p$.*

(4.3 $\Longrightarrow$ 4.4) *According to equation 4.3 there is a total order amongst the languages. In the case that $i \leq j$ we have $L_i \prec L_j$, further, $L_i \prec L_i$ holds because $i \leq i$. Therefore, the family of languages property holds for $L' = L_j$. In the case that $j \leq i$ we can show that the family of languages property can be established analogously for $L' = L_i$.*

From a practical point of view, the fact that we can find a language every other language can be translated to is not sufficient to facilitate information sharing, because we have no guarantee that we will still be able to draw inferences from the integrated models, because it is well known that there are certain combinations of logical operators that lead to undecidability of a language. We therefore claim that every integrated language has to be decidable. This weakens the claims made in section 3.3 where we demanded the existence of a reasoner for the integrated language. For practical applications this claim still holds, from a theoretical point of view, however, decidability is the most important property, because it is independent of a specific reasoner. Therefore we define guarded family of languages as follows:

Definition 4.3 (Guarded Family) *A guarded family of languages is a family $\mathcal{L}$ where for every pair of languages L_1, L_2 with $(L_1 \prec L') \wedge (L_2 \prec L')$ satisfiability reasoning in the language L' is decidable.*

The literature on terminological languages already discusses sets of languages with a certain internal structure. Figure 4.1 shows the family of languages known as the $\mathcal{AL}$ and $\mathcal{FL}^-$ Hierarchy of description logics as it has been investigated by [Kurtonina and de Rijke, 1999]. Using the notions introduced in this section this hierarchy is very similar to a family of terminological languages.

In order to compare to compare the relative expressiveness of languages [Kurtonina and de Rijke, 1999] use the notion of a bi-simulation which is used

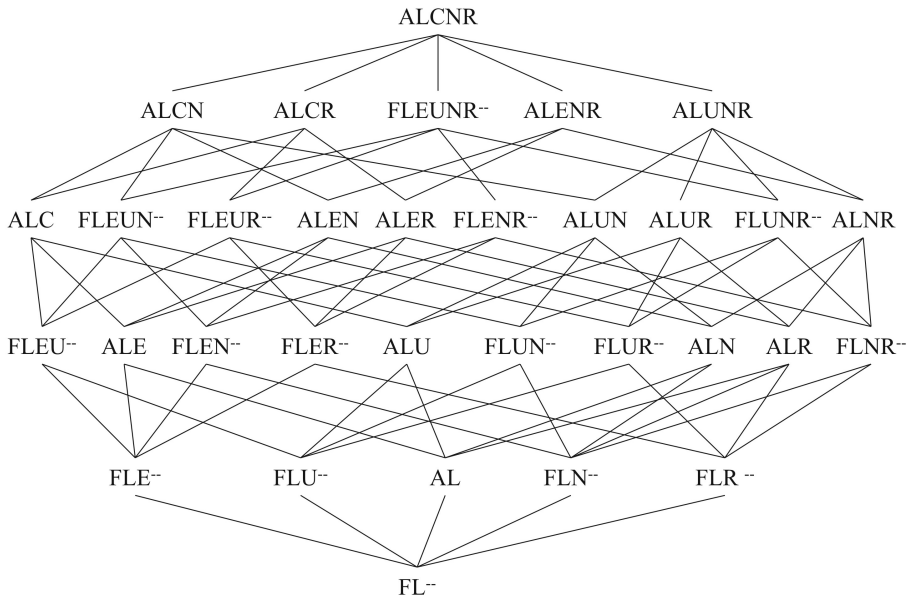


Figure 4.1: The $\mathcal{AL}$ and $\mathcal{FL}^-$ Hierarchy of Description logics. (taken from [Kurtonina and de Rijke, 1999])

to define a structure between the languages. In the following, we will use local transformations with certain properties to define such structures that can directly be used in order to perform the transformations. We consider our approach more practical, because transformations will be used for knowledge integration in any ways and should therefore be used as a structuring principle in order to receive results that can be directly used in applications.

4.2 Terminological Languages

In order to further investigate the notion of coverage between ontology languages, we have to define a general framework for specifying information semantics in terms of explicit models of ontological knowledge that is compatible with the language standard RDF schema and DAML+OIL. We argue that such a framework should be based on a class of logics known as description logics [Donini et al., 1996]. The rational for this is the following:

- The expressiveness and complexity of these languages has been studied thoroughly and well-founded results are available [Donini et al., 1991, De Giacomo, 1995]
- It has been shown that description logics provide a unifying framework

for many class-based representation formalisms [Calvanese et al., 1999].

- Description logic-based languages have become of interest in connection with the semantic Web. The language OIL [Fensel et al., 2000] which had a major influence on the development of DAML+OIL [van Harmelen et al., 2001] is a prominent example.

Further description logics completely satisfy our claim for flexibility, because we already get a notion of languages in terms of special logics. These logics result from the combination of operators. One of the most well known patterns is $\mathcal{ALC}$ the description logic containing Boolean operations on class expressions as well as universal and existential restrictions on slot fillers. The pattern used to resemble different class-based representation formalisms in [Calvanese et al., 1999] is $\mathcal{ALUNI}$ which contains the following operators: conjunction, disjunction, negation, universal restrictions on slot fillers, number restrictions and inverse slots. $\mathcal{SHIQ}$, the logic underlying OIL also contains existential restrictions, transitive slots and conjunction of slot definitions. Theoretical results from the field of description logics provide us with the knowledge about decidable combinations of modeling primitives and their complexity with respect to subsumption reasoning. Consequently, every decidable combination of operators is a potential language that can be used to build the ontology for a certain application.

In the following, we describe a general definition of terminological languages based on description logics introduced in [Baader et al., 2000] and show that DAML+OIL can be regarded as an instantiation of the general framework.

4.2.1 A General Framework

A terminological language mainly consists of operators that can be used to form complex class definitions from simple ones. In the following we give an abstract definition of terminological languages.

Definition 4.4 (Terminological Language [Baader et al., 2000]) *A terminological language L is the set of legal expressions e over a set T of atomic terms and a set F of operators where expressions are recursively defined as follows:*

- *every $t \in T$ is a legal expression*
- *if e is a legal expression, then $\neg e$ is also a legal expression*
- *it e_1 and e_2 are legal expressions, then $e_1 \wedge e_2$ and $e_1 \vee e_2$ are legal expressions*
- *if $f \in F$ in an n-ary operator and $e_1, \cdots, e_n$ are legal expressions then $f(e_1, \cdots, e_n)$ is a legal expression*

Note the set that T of atomic terms is independent of a specific language.

Each class in an ontology can now be intentionally described by a legal expression. The intended meaning of the expression, and therefore the class, can be described using an interpretation mapping assigning elements from an abstract domain to a class, thereby providing model-based semantics. In order to define models of terminological expression, we use the abstract description model proposed by Baader and others:

Definition 4.5 (Abstract Description Model [Baader et al., 2000]) *An abstract description model is of the form:*

$$\Im = \langle W, F^{\Im} = (f_i^{\Im})_{i \in I} \rangle$$

where W is a nonempty set and $f_i^{\Im}$ are functions mapping every sequence $\langle X_1, \cdots, X_{n_i} \rangle$ of subsets of W to a subset of W.

Equipped with the notion of model, we can define the interpretation mapping in two steps. First we assume an assignment $\mathcal{A}$ mapping every $t \in T$ to a subset of W, then we define the interpretation mapping recursively as follows:

Definition 4.6 (Language Semantics [Baader et al., 2000]) *Let L be a terminological language and $\Im = \langle W, F^{\Im} \rangle$ an abstract description model. An assignment $\mathcal{A}$ is a mapping from the set of atomic terms T to 2^W. The assignment of a subset of W to a term t is denoted by $t^{\mathcal{A}}$. The Interpretation $\delta^{\Im,\mathcal{A}}$ of an expression $\delta \in L$ is now defined by:*

1. $t^{\Im,\mathcal{A}} := t^{\mathcal{A}}$ *for every* $t \in T$
2. $(\neg e)^{\Im,\mathcal{A}} := W - e^{\Im,\mathcal{A}}$
3. $(e_1 \wedge e_2)^{\Im,\mathcal{A}} := e_1^{\Im,\mathcal{A}} \cap e_2^{\Im,\mathcal{A}}$
4. $(e_1 \vee e_2)^{\Im,\mathcal{A}} := e_1^{\Im,\mathcal{A}} \cup e_2^{\Im,\mathcal{A}}$
5. $f(e_1, \cdots, e_n)^{\Im,\mathcal{A}} := f^{\Im}(e_1^{\Im,\mathcal{A}}, \cdots, e_n^{\Im,\mathcal{A}})$ *for every* $f \in F$

The semantics definition given above is the basis for deciding whether a class definition is equivalent, more specialized or more general than another. Formally, we can decide whether one of the following relations between two expressions holds:

subsumption: $e_1 \sqsubseteq e_2 \iff e_1^{\Im,\mathcal{A}} \subseteq e_2^{\Im,\mathcal{A}}$

equivalence: $e_1 \equiv e_2 \iff (e_1 \sqsubseteq e_2 \wedge e_2 \sqsubseteq e_1)$

These two properties allow us to conclude that a Web page belonging to the class defined by e_1 also belongs to the class defined by e_2. In the following we will make use of this observation in order to deal with classification hierarchies.

4.2.2 DAML+OIL as a Terminological Language

The framework described above covers a wide range of logical languages including some modal logics and description logics. As DAML+OIL is based on a variant of description logics, the operators of DAML+OIL can be interpreted in the framework of terminological languages.

Concept Forming Operators: In order to define DAML+OIL in the general framework, we first have to identify the set of operators $F_{DAML+OIL}$. The mapping from DAML+OIL properties to concept forming operators of an expressive description logic depicted in figure 4.2. Note that general notion of terminological language does not contain explicit notions of relations. Therefore, we introduce the set of operators that include a functions for every relation included in the model including inverse relations.

DAML+OIL Property	**Terminological Operator $f^{\mathcal{I}}$**
`daml:intersectionOf`	$C_1 \sqcap \cdots \sqcap C_n$
`daml:unionOf`	$C_1 \sqcup \cdots \sqcup C_n$
`daml:complementOf`	$\neg C$
`daml:oneOf`	$\{x_1, \cdots, x_n\}$
`daml:toClass`	$\forall P.C$
`daml:hasClass`	$\exists P.C$
`daml:hasValue`	$\exists P.\{x\}$
`daml:minCardinalityQ`	$\leq nP.C$
`daml:maxCardinalityQ`	$\geq nP.C$
`daml:cardinalityQ`	$= nP.C$

Figure 4.2: Terminological Operators of DAML+OIL [Horrocks, 2001]

Beside concept forming operators, DAML+OIL also contains properties that introduce explicit subsumption and equivalence relations between concepts and pairs of concepts that are in a certain relation. Figure 4.3 depicts the mapping from DAML+OIL properties to terminological axioms. Together with the operators in figure 4.2 these operators capture the semantics of DAML+OIL. Operators not mentioned in one of the mappings, e.g. simple number restrictions can be simulated using other DAML+OIL operators.

Model-Based Semantics: In order to define the semantics of DAML+OIL, we use an abstract description model as given in definition 4.5. We choose W to be the set of all objects that might occur in a domain. In order to complete the model, for each function $f^{\mathcal{I}} : W \rightarrow W$ we have to define function value $f^I(X)$. Once the value is defined for every function definition 4.6 provides us with a model-theoretic semantics of the language. Figure 4.4 defines the

DAML+OIL Property	Terminological Axiom
`rdf:type`	$x : C$
`rdfs:subClassOf`	$C_1 \sqsubseteq C_2$
`daml:sameClassAs`	$C_1 \equiv C_2$
`daml:disjointWith`	$C_1 \sqsubseteq \neg C_2$
`daml:sameIndividualAs`	$\{x_1\} \equiv \{x_2\}$
`daml:differentIndividualFrom`	$\{x_1\} \sqsubseteq \neg\{x_2\}$
`daml:subPropertyOf`	$P_1 \sqsubseteq P_2$
`daml:samePropertyAs`	$P_1 \equiv P_2$
`daml:inversOf`	$P_1 \sqsubseteq P_2^-$
`daml:transitiveProperty`	$P^+ \sqsubseteq P$
`daml:uniqueProperty`	$\top \sqsubseteq \leq 1P$
`daml:unambigousProperty`	$\top \sqsubseteq \leq 1P^-$

Figure 4.3: Terminological Axioms of DAML+OIL [Horrocks, 2001]

function values for the terminological operators used in DAML+OIL adapted from the KRSS specification [Patel-Schneider and Swartout, 1994]. Again note that for all operators referring to a relation we have one operator for every relation in the model. In order to define the semantics of operators that incorporate a role name, we have to add an interpretation function for role names [Baader et al., 2000]. The interpretation of a relation P denoted as $R^{\mathcal{I}}$ is defined as a function into $W \times W$.[1]

Terminological Operator $f^{\mathcal{I}}$	Extensional Model $f^{\mathcal{I}}(X)$
$C_1 \sqcap \cdots \sqcap C_n$	$C_1^{\Im,\mathcal{A}} \cap \cdots \cap C_n^{\Im,\mathcal{A}}$
$C_1 \sqcup \cdots \sqcup C_n$	$C_1^{\Im,\mathcal{A}} \cup \cdots \cup C_n^{\Im,\mathcal{A}}$
$\neg C$	$W - C^{\Im,\mathcal{A}}$
$\{x_1, \cdots, x_n\}$	$\{x_1, \cdots, x_n\} \subset \mathcal{W}$
$\forall P.C$	$\{y \in W \vert (y,x) \in P^{\mathcal{I}} \implies x \in C^{\Im,\mathcal{A}}\}$
$\exists P.C$	$\{y \in W \vert \exists x((y,x) \in P^{\mathcal{I}}) \wedge x \in C^{\Im,\mathcal{A}}\}$
$\exists P.\{x\}$	$\{y \in W \vert (y,x) \in P^{\mathcal{I}}\}$
$\leq nP.C$	$\{y \in W \vert \ \vert\{x \vert (y,x) \in P^{\mathcal{I}} \wedge x \in C^{\Im,\mathcal{A}}\}\vert \leq n\}$
$\geq nP.C$	$\{y \in W \vert \ \vert\{x \vert (y,x) \in P^{\mathcal{I}} \wedge x \in C^{\Im,\mathcal{A}}\}\vert \geq n\}$
$= nP.C$	$\{y \in W \vert \ \vert\{x \vert (y,x) \in P^{\mathcal{I}} \wedge x \in C^{\Im,\mathcal{A}}\}\vert = n\}$

Figure 4.4: Semantics of DAML+OIL Operators

[1] In a later version of the paper Baader and others abstract description models directly introduce an interpretation of relation [Baader et al., 2002]. This change does not affect the results described in the following.

The semantics of the terminological axioms in figure 4.3 is mainly given by the definition of the subsumption an the equivalence relation in the last section. Using the interpretation of role names the notion of subsumption in section 4.2.1 can be directly transferred to roles. The notion of role equivalence is defined analogously to concept equivalence. In order to define transitive roles we define P^+ as follows:

$$P^{+\,\mathcal{I}} \iff_{def} \{(x,y)|(x,z) \in P^{\mathcal{I}} \wedge (z,y) \in P^{\mathcal{I}}\} \tag{4.5}$$

Finally, the inverse relation P^- is defined as follows:

$$P^{-\,\mathcal{I}} \iff_{def} \{(x,y)|(y,x) \in P^{\mathcal{I}}\} \tag{4.6}$$

The semantics of the terminological axioms in DAML+OIL that use these additional definitions is summarized in figure 4.5.

The definition of the semantics of DAML+OIL in terms of the general framework for terminological languages does not only provide the basis for logical inference, it also enables us to relate the DAML+OIL language to other terminological languages that can be used to define a controlled vocabulary for semantics-preserving information exchange. We also see that the general notion of terminological languages fulfills our requirements for encoding controlled languages as it can be used to specify the emerging standards for knowledge modeling on the Web and are highly flexible. We will therefore build our further work on the framework presented in this section.

Terminological Axiom	**Interpretation**
$x : C$	$x \in C^{\Im,\mathcal{A}}$
$C_1 \sqsubseteq C_2$	$C_1^{\Im,\mathcal{A}} \subseteq C_2^{\Im,\mathcal{A}}$
$C_1 \equiv C_2$	$C_1^{\Im,\mathcal{A}} = C_2^{\Im,\mathcal{A}}$
$C_1 \sqsubseteq \neg C_2$	$C_1^{\Im,\mathcal{A}} \cap C_2^{\Im,\mathcal{A}} = \emptyset$
$\{x_1\} \equiv \{x_2\}$	$\{x_1\}^{\Im,\mathcal{A}} = \{x_2\}^{\Im,\mathcal{A}}$
$\{x_1\} \sqsubseteq \neg\{x_2\}$	$\{x_1\}^{\Im,\mathcal{A}} \neq \{x_2\}^{\Im,\mathcal{A}}$
$P_1 \sqsubseteq P_2$	$P_1^{\mathcal{I}} \subseteq P_2^{\mathcal{I}}$
$P_1 \equiv P_2$	$P_1^{\mathcal{I}} = P_2^{\mathcal{I}}$
$P_1 \sqsubseteq P_2^-$	$P^{\mathcal{I}} \subseteq P^{-\,\mathcal{I}}$
$P^+ \sqsubseteq P$	$P^{+\,\mathcal{I}} \subseteq P^{\mathcal{I}}$
$\top \sqsubseteq \leq 1P$	$\|\{y\|(x,y) \in P^{\mathcal{I}}\}\| = 1$
$\top \sqsubseteq \leq 1P^-$	$\|\{x\|(x,y) \in P^{\mathcal{I}}\}\| = 1$

Figure 4.5: Semantics of Terminological Axioms

4.3 Families of Terminological Languages

The family of languages property (definition 4.4) may apply to various kinds of languages and arbitrary transformations between these languages. In this section, we discuss the property with respect to terminological languages as introduced in section 4.2. Based on the general definition of these languages we discuss the coverage relation $\prec$ and define it by transformations that preserve formal properties of the transformed models. We investigate transformations that preserves interpretation and models and discuss transformations that preserve consistency of concept expressions.

In the following, a language L will be a set of legal expressions. A representation (r) is a set of expressions in L. Further, by F_L we refer to the set of operators allowed in language F (compare definition 4.4).

4.3.1 Language Inclusion

We first consider the simplest case where one language is completely included in the other. The definition of a language as a set of expressions enables us to define language inclusion by set inclusion:

Definition 4.7 (Language inclusion) *A language L is included in another language L' iff $L \subseteq L'$.*

While this case seems to be trivial, it has still practical relevance, because many languages are defined as extensions of other languages. We give an example from the field of description logics.

Example 1 (The FaCT Reasoner) *The FaCT description logic reasoner implements two reasoning modules one for the language $\mathcal{SHF}$ and one for the language $\mathcal{SHIQ}$ which simply extends $\mathcal{SHF}$ with inverse roles and qualified number restrictions. As a consequence, $\mathcal{SHF}$ models can be handled by the $\mathcal{SHIQ}$ reasoner without change.*

The notion of language coverage gives us a first criterion to define a special kind of coverage relation that can be used in order to find a language that covers languages to be integrated.

Definition 4.8 (Language-based Coverage)

$$L \leqq L' \Leftrightarrow_{def} (L \subseteq L')$$

In order to use language based coverage for language integration we have to test whether one language is a subset of another. The subset criterion given in definition 4.7 is not very suitable for implementing a test procedure. We therefore use the correspondence between logical operators and the set of legal expressions in order to define a criterion that based on the operators of a terminological language:

Proposition 2 *A language L includes another language L' syntactically ($L' \subseteq L$) iff $F_{L'} \subseteq F_L$.*

We prove the correspondence between operators and legal expressions using definition 4.4.

Proof 2 *($\Rightarrow$) We show that $\delta \in L' \implies \delta \in L$ by induction over the definition of δ in L'. If δ is a term then $\delta \in L$, because the definition of L is independent of T. In the induction step we assume that $\delta = f(\delta_1, \cdots, \delta_n)$, where $f \in F_{L'}$ and $\delta_1, \cdots, \delta_n \in L$ (by induction hypothesis). We know that also $f \in F_L$, because $F_{L'} \subseteq F_L$. From the fourth bullet in definition 4.4 it follows that $\delta \in L$.*

($\Leftarrow$) We show that $f \in F_{L'} \implies f \in F_L$ by assuming that there is an operator $f \in F_{L'}$ such that $f \notin F_L$ and establishing a contradiction: If $f \in F_{L'}$ then $f(t_1, \cdots, t_n)$ is a legal expression in L' for legal expressions $\delta_1, \cdots, \delta_n$. As $L' \subseteq L$ we get $f(t_1, \cdots, t_n)$ is a legal expression in L. This contradicts with $f \notin F_L$, because $f(t_1, \cdots, t_n)$ is legal in L only if $f \in F_L$.

Using this results, we can define a family of languages by the maximal set of operators that may appear in the languages that belong to the family.

Definition 4.9 (Operator-based Family of Terminological Languages) *Let $\mathcal{F}$ be a set of operators then the family $\mathcal{L}_\mathcal{F}$ of terminological languages defined by $\mathcal{F}$ is the set of Languages $\{L_i | F_{L_i} \subseteq \mathcal{F}\}$.*

The definition of the family in terms of a set of all subsets of operators implies that the languages in an operator-based family form a lattice. From lattice theory, we get that the family of languages property is satisfied for operator-based families of languages.

Proposition 3 *Every family of terminological languages $\mathcal{L}$ satisfies the family of languages property in equation 4.4 for $\prec = \leqq$*

The proposition can be proven in a straightforward way:

Proof 3 *If $L_1, \cdots, L_n$ are the languages to be integrated, we chose L to be the language defined by $F_L = \bigcup_{i=1}^{n} F_{L_i}$. From set theory we obtain $F_{L_i} \subseteq F_L$ for $i = i, \cdots, n$. Proposition 2 yields that $L_i \leqq L$ which establishes the family of languages property.*

Operator-based families of terminological languages already come with a number of advantages. We can build upon a rich set of terminological operators and choose only the fraction we need to define the semantics of different information repositories. Using the notion of a family of languages, we can integrate the semantic models of different information sources on the language even if they do not use the same set of operators as long as they are in the same family.

A problem that persists is the size of the language needed to encode integrated knowledge. The more languages are to be integrated the more operators will have to be in the integrated language. We will soon end up with an approach that is very close to the pivot approach where the pivot language is the one that contains all operators of the family. In order to avoid this problem we have to consider alternative notions of coverage between languages.

4.3.2 Interpretation Preservation

In order to reduce the number of operators used to encode the integrated language we use the possibility to encode logical operators using a combination of other operators. This possibility applies to single operators (encode disjunction by conjunction and negation) as well as to combination of operators (see example below). In order to make use of this kind of transformations while still guaranteeing that the semantics of transformed models we demand that a transformation preserves interpretation of any given model.

Definition 4.10 (Interpretation preservation) *A transformation τ preserves the interpretations iff*

$$\forall \delta \in L, \forall \Im, \mathcal{A} \ \ (\delta^{\Im,\mathcal{A}} = \tau(\delta)^{\Im,\mathcal{A}})$$

Interpretation preserving transformations are used in practical applications in oder to translate from controlled languages like OIL into terminological languages that are supported by inference engines. A concrete example of an interpretation-preserving transformation is the following:

Example 2 (Reasoning in Core-OIL) *The lowest layer of the ontology language OIL which has gained significant attention in connection with the semantic Web is Core -OIL which provides a formal semantics for a part of RDF schema. In order to provide reasoning services, the language is translated into the logic $\mathcal{SHIQ}$ and the FaCT Reasoner is used to provide the reasoning services [Horrocks, 1998]. The translation contains the following interpretation-preserving mapping:*

$$\tau(R \sqsubseteq (\texttt{domain}\, C)) = \top \sqsubseteq (\texttt{all}\, (\texttt{inv}\, R)\, C)$$

Where C is a class expression and R a relation name.

Based on interpretation-preserving transformations, we define a notion of coverage that is based on the semantics of a representation instead of the operators that are used.

Definition 4.11 (Interpretation-based Coverage)

$L \lessapprox L' \Leftrightarrow_{def}$ there exists an interpretation preserving transformation $\tau : L \rightarrow L'$

Using this weaker notion of coverage we can extend the definition of a family of languages to include such languages that can be transformed into a language of the family. This new notion may include languages with operators that are not included in the family. The definition of a terminological language makes it possible to define semantic coverage in a modular way. We only require that for each operator not included in the family definition, there has to be an interpretation-preserving transformation into a combination of operators that are in the family for every operator not included:

Proposition 4 *A language L covers a language L' according to definition 4.11 if for every* $f \in F_{L'}, f \notin F_L$ *and every legal expression of the form* $\delta_{L'} = f(\delta_1, \cdots, \delta_n)$ *there exist* $f_1, \cdots, f_m \in F_L$ *such that* $\delta_L^{\Im,\mathcal{A}} = \delta_{L'}^{\Im,\mathcal{A}}$ *and* δ_L *is a legal expression over T and* $F = \{f_1, \cdots, f_m\}$.

We can prove that local transformations indeed preserve semantics in the following way:

Proof 4 *We prove that for every expression* $\delta \in L$ *with* $\Im, \mathcal{A} \models \delta$ *also* $\Im, \mathcal{A} \models \tau(\delta)$*, where* $\tau(\delta)$ *is achieved by replacing unknown operators in the way described in proposition 4.*

If $\delta = t$ *for a term* $t \in T$ *and* $\Im, \mathcal{A} \models_L t$ *then* $\Im, \mathcal{A} \models_{L'} t$ *also holds, because the same assignment mapping is used and the term is mapped on the same subset of W in both languages. In the induction step we consider* $\delta = f(\delta_1, \cdots, \delta_n)$*. By the induction hypothesis we get that* $\Im, \mathcal{A} \models_{L'} \delta_1, \cdots, \delta_n$*. More specifically,* $\delta_1, \cdots, \delta_n$ *are mapped on the same subsets of W as in L.*

Case 1: f *is also in* $F_{L'}$*. Then in* $\delta_{L'}$ *is mapped on the same subset as* δ_L *as the interpretation of* f *is the same and* $f^{\Im}$ *is applied to the same subsets of W (induction hypothesis). Therefore* $\Im, \mathcal{A} \models_{L'} \delta_{L'}$ *holds.*

Case 2: f *is not in* $F_{L'}$*. In this case* δ_L *is replaced by* $\delta_{L'}$*. From proposition 4 we get that both expressions are mapped on the same subset of W. Therefore* $\Im, \mathcal{A} \models_{L'} \delta_{L'}$ *holds.*

Equipped with the result that local transformations preserve global interpretations, we now formally define model-based families of terminological languages in the following way:

Definition 4.12 (Model-based Family of Terminological Languages) *Let* $\mathcal{F}$ *be a set of operators then the family* $\mathcal{L}_{\mathcal{F}}$ *of terminological languages defined by* $\mathcal{F}$ *is the set of Languages* $Syn_{\mathcal{L}} \cup Sem_{\mathcal{L}}$ *where* $Syn_{\mathcal{L}} = \{L_i | F_{L_i} \subseteq \mathcal{F}\}$ *and* $Sem_{\mathcal{L}} = \{L_j | \exists L \in Syn_{\mathcal{L}}$ *such that* $L_j \lessapprox L\}$

The definition of a model-based family of terminological languages ensures that the family of languages property holds for these family if we use the notion of interpretation-based coverage.

Proposition 5 *Every model-based family of terminological languages as defined in definition 4.12 above fulfills the family of languages property (equation 4.4) for* $\prec = \lessapprox$.

Proof 5 *Let* $L_1, \cdots, L_n$ *be the languages to be integrated. If all languages are in* $Syn_{\mathcal{L}}$ *we can apply proposition 3 to establish the result. In the following, we consider the case where there are languages* $L_{i_1}, \cdots, L_{i_k}$ *that are in* $\mathcal{L} - Syn_{\mathcal{L}}$.

By definition, for each language L_i *there is a language* $L_i' \in Syn_{\mathcal{L}}$ *such that* $L_i \lessapprox L_i'$. *From proposition 3 we get that there is an integration language* L *if we replace the languages* $L_{i_1}, \cdots, L_{i_k}$ *by* $L_{i_1}', \cdots, L_{i_k}'$. *In order to prove the proposition, we thus have to show that* $(L \leqq L_i' \lessapprox L_i) \implies (L \lessapprox L_i)$ *for every* L_i *that has been replaced.*

It is sufficient to show that $L \leqq L' \implies L \lessapprox L'$, *because* $\lessapprox$ *is transitive. We prove this by induction over the definition of expressions* $\delta \in L$. *If* δ *is an atomic term* $t \in T$ *it is mapped on the same subset of* W, *because the assignment mapping* $\mathcal{A}$ *is independent of the language. If* δ *is a complex expression* $f(\delta_1, \cdots, \delta_n)$ *it will also be mapped on the same subset of* W, *because*

1. δ_i *will be mapped on the same subset by induction hypotheses*
2. f *is contained in* $F_{L'}$ *by premise*
3. *The interpretation of* f *is the same, because the same interpretation mapping is applied.*

The extended notion of a family of languages enables us to integrate a wide range of terminological languages without having to use the operators of all integrated languages. However, the approach is restricted in the sense that there are of course operators that cannot be completely encoded using other operators. Using only transformations that preserve interpretation, we will often end up with a combination of operators that are hard to handle in reasoning. The use of instances in concept definitions for example is inherently intractable [Schaerf, 1994], especially if inverse roles are used. If we still want to reason about models encoded in these languages, we have to use transformations with weaker formal properties.

4.3.3 Weakening Transformations

The applicability of semantics-preserving transformations is somewhat restricted, because we can only expect to find such transformations into languages that are at least as expressive as all languages we translate from. However, if we want to scale up reasoning we have to restrict the expressive power of representation languages in favor of efficient reasoning. We therefore also consider transformations that imply a loss of information. These transformations can still be useful, because they sometimes preserve some formal properties that support reasoning with the transformed models. In the following we discuss such properties and give examples for corresponding transformations.

Consequence preservation

Consequence preservation can be considered the ultimate goal of semantic interoperability: it denotes the fact that the consequences (what is true in all models) of the source and the target representations are the same (modulo transformation). We can define consequence preservation using the consistency criterion. This is possible, because in order to decide whether an expression follows from another it is sufficient to show that the conjunction of the first expression together with the negation of the second is inconsistent. Using this fact, we can define consequence preserving transformations as follows:

Definition 4.13 (Consequence preservation) *A transformation τ is said consequence-preserving iff*

$$\forall\delta, \forall\Im, \mathcal{A} \ (\delta^{\Im,\mathcal{A}} \neq \emptyset \iff \tau(\delta)^{\Im,\mathcal{A}} \neq \emptyset)$$

This kind of transformation is very relevant in practice, because it enables us to change the representation of knowledge without losing the ability to derive implicit knowledge from the model which is the major benefit of using a logical rendering of a controlled language. We give an example of a consequence-preserving transformation from the area of intelligent information integration.

Example 3 (Deciding Query Containment:) *Calvanese and others describe an approach for a formalization of database schema and queries as well as a procedure for deciding query containment [Calvanese et al., 1998a]. The basis for the method is $\mathcal{DLR}$, a description logic with n-ary relations. In order to implement the approach, Horrocks and others introduce a translation from $\mathcal{DLR}$ to $\mathcal{SHIQ}$ (see above). In [Horrocks et al., 1999] it is shown, that the transformation has exactly the property described in definition 4.13.*

The practical benefits of consequence preserving transformations imply that we also use these transformation in order to define the coverage relation used in the language customization process.

Consistency preservation

An even weaker property to be considered is consistency preservation. This property is important in order to ensure that a transformed knowledge base does not become inconsistent making reasoning impossible. Further, consistency preservation guarantees that inference in the transformed model based on refutation is still correct. We cannot guarantee completeness, but at least no wrong conclusions are drawn.

Definition 4.14 (Consistence preservation) *A transformation τ is said to be inconsistency-preserving iff*

$$\forall\delta, \forall\Im, \mathcal{A} \ (\delta^{\Im,\mathcal{A}} \neq \emptyset \implies \tau(\delta)^{\Im,\mathcal{A}} \neq \emptyset)$$

A practical example for the application of consistency preserving transformation is the use of individuals in concept definitions. Reasoning with individuals has been shown to be inherently intractable [Schaerf, 1994]. One solution to provide tractable inferences is to use weakening transformations like in the example given below:

Example 4 (Reasoning in Instance-OIL) *The second layer of the OIL language called standard OIL provides an expressive language for building ontologies. Again, the language is translated into $\mathcal{SHIQ}$ in order to provide inference services. Standard OIL also includes capabilities for expressing assertional knowledge and instances in concept definitions. As the FaCT reasoner does not support instance reasoning, the translation from Standard OIL to $\mathcal{SHIQ}$ includes some mappings that do not preserve the complete semantics, but preserve consistency, e.g.*

$$\tau((\texttt{one} - \texttt{of}\ i_1\ i_2)) = (\texttt{or}\ I_1\ I_2)$$

This transformation replaces the enumeration of instances by a disjunction of concepts with the same name.

We also include consistency preserving transformations into the set of properties for defining the coverage relation, because we found practical examples where consistency preserving transformations are used that are not complete with respect to reasoning.

4.4 Implementing the Family

The transformations discussed above do not only serve a theoretical purpose, i.e. characterizing the semantic properties of a transformed knowledge base, they can also be implemented in order to really execute the transformations between different kinds of logics. We implement the different kinds of relations in the framework of proof-carrying transformation described by Euzenat [Euzenat, 2001] using the standards XML and XSLT.

4.4.1 Modular Encoding

The basis for an implementation of the family of languages approach is an XML encoding of the terminological languages we want to include in our families. We have shown above that the most interesting families, i.e. operator-based and model-based families are established by local transformations that replace single operators in a language by an expression formed of operators in another logic. In order to support these local transformations, a modular encoding known as DLML of terminological languages based on single operators is used [Euzenat, 2001].

The encoding of a terminological language consists of three basic elements:

Atoms for representing concept and role names

Operators that are used to form regular expressions

Formulae constructors used to state terminological axioms.

The most important elements are the operator definitions, because they are modified during transformation processes. Following the idea of providing transformations with provable properties, the encoding of an operator consists of two parts. The first one is a DTD describing in which syntactical form the operator should be represented in the language. The inverse role operator (see section 4.2.2) for example will be defined as follows:

```
<!ELEMENT dl:INV (%dl:RDESC;)>
```

In addition to the syntactic definition, DLML also provides an explicit representation of the semantics of a constructor. To be more specific, it describes the interpretation $f^{\mathcal{I}}$ of an operator. The mathematical markup language MML that provides useful mathematical primitive is used to specify this interpretation. For the case of the inverse operators the semantics definition also called DSD is the following:

```
<dsd:DSD> <dsd:denotation match="dl:INV">
     <mml:eq/>
        <mml:apply>
           <mml:inverse/>
           <!-- converse for binary relations -->
           <dsd:apply-interpretation select="*[1]"/>
        </mml:apply>
</dsd:denotation> </dsd:DSD>
```

In this way DLML specifies about 40 different terminological operators. In order to describe an entire language in terms of its operators, the definitions of operators are included into a single DTD that now describes the syntactic structure of a terminological language. The language $\mathcal{SHIQ}$ for example that forms the semantic basis of the OIL language is defined as follows:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?> <!DOCTYPE
dlml:logic SYSTEM
     "http://co4.inrialpes.fr/xml/dlml/logic/dlml.dtd">

<dlml:logic name="shiq" version="1.0">
           <dlml:atoms/>
           <dlml:cop name="anything"/>
           <dlml:cop name="nothing"/>
           <dlml:cop name="and"/>
           <dlml:cop name="or"/>
           <dlml:cop name="not"/>
           <dlml:cop name="all"/>
           <dlml:cop name="some"/>
           <dlml:cop name="csome"/>
```

```
          <dlml:cop name="catleast"/>
          <dlml:cop name="catmost"/>
          <dlml:rop name="inv"/>
          <dlml:rop name="trans"/>
          <dlml:cint name="cprim"/>
</dlml:logic>
```

The definitions contained in DLML provides us with a unique syntactical basis that is needed for applying XSLT transformation to the problem of converting models from one logical language into another.

4.4.2 Transformations

Based on the XML encoding of terminological languages, we can define translations in term of XSLT transformations between operator definitions. We exemplify this possibility using the example of a weakening transformation from example 4. The corresponding XSLT transformation that replaces each occurrence of the `one-of` operator by a conjunction of newly introduced classes declared to be disjoint is the following:

```
<xsl:template match="dl:TERMINOLOGY">
     <dl:TERMINOLOGY>
       <xsl:comment>
          The terminology
       </xsl:comment>
       <xsl:apply-templates />
       <xsl:comment>
          Introduction of the ONEOF (optional)
       </xsl:comment>
       <xsl:apply-templates mode="gatheroneof" />
     </dl:TERMINOLOGY>
</xsl:template>

<!-- usual processing -->

<xsl:template match="*|@*|text()">
     <xsl:copy>
       <xsl:apply-templates select="*|@*|text()"/>
     </xsl:copy>
</xsl:template>

<xsl:template match="dl:ONEOF">
     <dl:OR>
       <xsl:for-each select="dl:INDIVIDUAL">
         <dl:CATOM>
           <xsl:value-of select="@name"/>
         </dl:CATOM>
       </xsl:for-each>
     </dl:OR>
```

```
</xsl:template>

<!-- gather one-of -->

<xsl:template match="dl:ONEOF" mode="gatheroneof">
     <xsl:for-each select="dl:INDIVIDUAL">
       <dl:CPRIM>
         <dl:CATOM>
          <xsl:value-of select="@name"/>
         </dl:CATOM>
         <dl:CATOM>
          <xsl:value-of select="dl:CATOM/text()" />
         </dl:CATOM>
       </dl:CPRIM>
     </xsl:for-each>
     <dl:CEXCL>
       <xsl:for-each select="dl:INDIVIDUAL">
         <dl:CATOM>
          <xsl:value-of select="@name"/>
         </dl:CATOM>
       </xsl:for-each>
     </dl:CEXCL>
</xsl:template>
```

The transformation generates a copy of the entire terminology. Whenever it encounters the `one-of` operator, it selects the individuals contained in the operator definition and creates a concept atom with the same name for each of these individuals. afterwards, concept definitions are created for these atoms in terms of terminological axioms and a statement is created that applies the disjointness operator (`dl:CEXCL`) is applied to the atoms and added to the new model. The result of the transformation is a terminological knowledge base, where the `one-of` operator has been replaced by the weaker notion of a disjoint conjunction of primitive concepts. Using the DSD of the operators involved, we can also prove that the transformation preserves consistency as stated above.

Conclusions

Transformations are an adequate means for ensuring interoperability on the language level, especially because they serve two purposes at the same time: transformations characterize the relation between two languages and at the same time describe a method for translating knowledge. We have seen that transformations operate in a modular way. They do not describe how a complete model has to be interpreted in a different language, but they define how specific languages elements have to be changed independently of other parts of the model. This modularity is a great advantage both from a theoretical and a practical perspective. With respect to the framework of terminological languages introduced in the last chapter, concept-building

operators provide the basis for defining local transformations. Transformations on the operator level provide great flexibility and they can even be tuned to a specific reasoning task. We have seen that there is not only the option of using transformations that completely preserve the semantics of a model. We can also define transformations that only preserve certain formal properties we need for different types of reasoning (e.g. consequence preservation for subsumption checking and consistency preservation for knowledge-base verification).

We conclude that the framework described above satisfies our requirements for representing and reasoning about ontologies. It is compatible with existing Web standards: it can be implemented using XML and XSLT (compare section 4.4) and we can also easily relate it to RDF schema and DAML+OIL by defining syntactic transformations from DLML (like the ones described in [Stuckenschmidt and Euzenat, 2001]). In especially, this is possible because the approach is semantically well founded through its relation to the notion of terminological language. This well-foundedness is not only beneficial for ensuring that a translation to DAML+OIL is correct and complete, it also allows us to perform terminological reasoning which is another requirement for a ontology representation approach.

Part II

Ontologies and Metadata

Chapter 5

Ontology Construction

In chapter 3, we discussed languages for explicating information semantics and argued for the need of an integration at the language level. In chapter 4 we adopted a general framework for characterizing such languages and showed that existing proposals can be captured by the general framework. We showed that we can formulate requirements on the relation between languages that enable us to compare models of information semantics. We now draw our attention to the nature and the content of ontologies needed to support information sharing. The goal is to define an architecture combining the advantages of global and local ontologies and to show how this infrastructure can be derived from an information sharing task.

Acknowledgement: Parts of this chapter have been published before. The description of the role of ontologies in the translation process is taken from [Visser et al., 2002] which is co-authored by Ubbo Visser, Gerhard Schuster and Thomas Voegele. The Example problem has first been discussed in [Stuckenschmidt and Visser, 2000] which is co-authored by Ubbo Visser. The strategy for building the infrastructure as well as its application to the example are taken from [Schuster and Stuckenschmidt, 2001] which is co-authored by Gerhard Schuster.

The acquisition of semantic knowledge has been identified to be a major bottleneck not only in information sharing but also in many other areas going back to expert system development. A whole scientific discipline called knowledge engineering is devoted to the task of providing tools and methods for support the knowledge acquisition and formalization process [Studer et al., 1998]. In connection with the interest in ontologies as a key technology to knowledge and information sharing the term 'ontological engineering' has become popular [Farquhar and Gruninger, 1997] and a number of methodologies for creating ontologies have been proposed [Gomez-Perez and Juristo, 1997, Uschold, 1996]. However, these methods are very general in nature as they aim at providing

general guidelines for all kinds of ontologies and purposes. We therefore propose a specialized strategy for the explication of information semantics.

In this chapter, we first discuss the problem of explicating information semantics in terms of a formal language on a general level and investigate critical aspects. Based on the investigation we develop an layered infrastructure for explicating information semantics across different information sources. Further, we describe and exemplify a customs tailored strategy for the development of the infrastructure on the basis of existing information sources and thesauri.

5.1 Ontologies and Knowledge Integration

Initially, ontologies have been introduced as a solution to the problem of explicating the semantics of formal languages used in knowledge sharing [Patil et al., 1991]. Later, it became clear that ontologies face the same or even harder problems with respect to heterogeneity as any other piece of information [Valente et al., 1999]. Presently, the problem of aligning different ontologies is considered to be one of the most challenging problems in knowledge representation. In order to avoid or at least reduce the problem of having to align heterogeneous ontologies when trying to integrate information sources we start with a careful investigation of the problem of explicating semantics using ontologies. The insights provided by this investigation will be the starting point for the design of an architecture for explicating information semantics.

As stated in the introduction, syntactic aspects are out of the scope of this thesis. The last chapter was devoted to interoperability of logical languages. In the remainder of the thesis, we will discuss the ontological layer. All attempts to provide generic solution to the alignment of the ontological dimension of a knowledge model have shown that this problem is in fact the most difficult one. In this section, we investigate the reason for the difficulties integration attempts face by comparing it with the other dimensions.

5.1.1 The Explication Dilemma

In AI the integration of knowledge is done on the basis of explicit representations of syntax, logical language and ontology. Integration of the syntactic languages is performed on the basis of a formal language that can be specified by grammar rules that form syntactic patterns. These patterns can be used to formulated transformation rules from one formal language into another. This is mainly done in mapping approaches like the OntoMorph [Chalupsky, 2000]. The logical dimension of a knowledge model is being addressed using a formal logic explicating admissible interpretations and consequences. We presented an approach for handling the logical dimension in the last chapter. Assuming a correct and complete inference engine able to work on the formal language

translations on the syntactic level can even be verified

If we now turn to the ontological dimension we notice that the explication of this additional dimension in line with the explication of the other dimensions has been promoted as a solution to the integration problem. However, it turns out that this explication leads to a serious problem because the explication of the ontological dimension of a knowledge model is again a knowledge model. Admittedly, this recursive approach has the advantages that we can address the question whether an ontological commitment is compatible with another on the logical level and even use deduction to prove this compatibility. The problem, however, is as the explication of the knowledge model is a knowledge model itself it has an ontological dimension. So, in order to decide whether the ontological dimension of two knowledge model is the same, in principle, we will also have to test whether the ontological dimension of the knowledge models that describe this dimension are the same and so on. We refer to this problem as the *explication dilemma*.

5.1.2 Avoiding the Explication Dilemma

In order to approach the explication dilemma it is obvious that we have to leave the recursion of defining the ontological dimension in terms of a knowledge model. In principle, the options we have is to reduce the problem of explicating the ontology either to the syntactic or the logical dimension. In the following, we will briefly discuss the possibilities and consequences of these two options.

Reducing Ontology to Syntax

The first option we have in order to avoid the explication dilemma is to reduce explication of the ontological dimension of a knowledge model to syntax. This means that rather than considering the ontological dimension in terms of a complete knowledge model we only use syntactic criteria to decide whether the ontological dimensions of two models are compatible or not. At first sight this solution seems not to be reasonable, but it is found frequently in knowledge integration. In description logics for example, atomic concepts are completely defined by their names and are therefore automatically assumed to have the very same intended meaning. Less restrictive ways of using the syntax in order to determine ontological equivalence are heuristic approaches of mapping between different concepts on a lexical or structural basis.

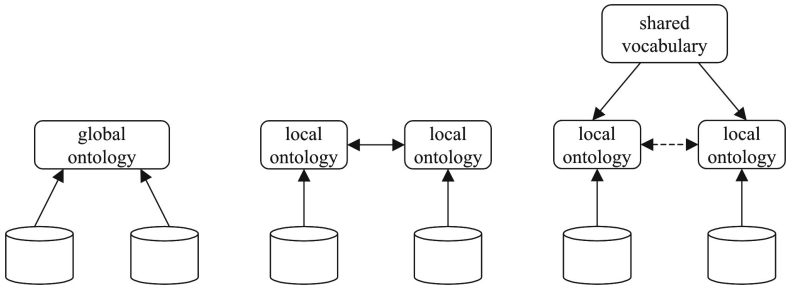
Reducing Ontology to Logic

The second option for avoiding the explication dilemma is to reduce the explication of the ontological dimension to the semantic dimension. This means that checking whether the ontological dimensions of two models are consistent to the question whether their axiomatizations share the same models. At first glance this seems to be a good solution, because the meaning of the model is taken

into account. However, there is also a loss of information compared to the original notion of ontology. A purely logical account of the ontological dimension does not take into account the intended meaning of names, because they are exchangeable as renaming does not affect the models of a logical theory. Therefore the valid interpretations we obtain by analyzing models will normally be much more than the interpretations intended by the underlying ontology that restricts these interpretations. Instead of considering the intended meaning, the reduction to logic is done by defining so-called interpretations. An interpretation is a translation from one logical theory into another that rephrases the axioms of the source theory using the signature from the target theory in such a way that all theorems are preserved. If such an interpretation exists, one can assume that the meaning is preserved. Relating this to ontologies it would mean that each term used in one knowledge model have to be definable using terms from other ontologies without losing consequences.

5.1.3 Integration Approaches

There are different options for relating ontologies using a reduction of ontology to syntax, logic or a combination of the two. The most often discussed ones are merging and mapping of ontologies [Klein, 2001]. We argue for an approach that is based on on-demand translations rather than a fixed connection of ontologies. The ideas of these approaches are depicted in figure 5.1.



(a) Merging (b) Mapping (c) Translation

Figure 5.1: Three approaches to relating ontologies

Merging aims at producing a single global ontology based on the ontologies to be related, thus reducing ontology to syntax. The integrated ontology will contain class definitions from both ontologies arranged in a single taxonomy. The advantage of this approach is that the merged ontology contains all information that is needed for information sharing. The drawback of this approach, however, is the effort needed to come up with a merged ontology that correctly resembles both ontologies. The situation becomes even worse if

we want to include further ontologies. In this case we have to perform further merges that are costly and lead to huge ontology that is hard to handle and maintain.

Mapping tries to avoid the creation of large global ontologies. It rather aims at relating concepts from both ontologies using additional knowledge, i.e. mapping rules. These rules can be used to share information, because the relation between semantic descriptions is made explicit using a reduction from ontology to logic. The advantage of the mapping approach is that an extension to other ontology does not affect the ones already included in the system. A problem of the mapping approach is the need to specify mappings to all other ontologies already present in a system. Further, static mapping rules can be come very complicated if not only one-to-one mappings are needed.

Translation is an approach that tries to combine the advantages of merging and mapping. The idea is to define concepts from different sources using terms specified in a shared model (similar to a partially merged ontology) and relate these concepts only if such a relation is needed using terminological reasoning. The advantages of the translation approach are its scalability - new information sources can be added without affecting other sources - and the fact that we do not have to establish a connection to each other source in the system. Mappings only have to be established with the shared model, because other relations are derived from these mappings.

5.2 A Translation Approach to Ontology Alignment

In weakly structured environments, it will frequently happen, that information sources are added or removed (compare definition 1.1). Further, if we consider environments like the World Wide Web, the number of information sources will be considerably high. Based on these observations, we conclude that the translation approach is most adequate for our purposes. Therefore, we will use the idea of integration by translation as a guideline when discussing the development of ontologies for information sharing.

5.2.1 The Translation Process

The proposed translation process is sketched below describing actors, supporting tools and knowledge items (i.e. ontologies) involved. Notice that although the approach described above translates only between two sources at a time, it is not limited to bilateral integration.

Authoring of Shared Terminology Our Approach relies on the use of a shared terminology in terms of properties used to define different concepts. This shared terminology has to be general enough to be used across all information sources to be integrated but specific enough to make meaningful definitions possible. Therefore the shared terminology will normally be built by an independent domain expert who is familiar with typical tasks and problems in a domain, but who is not concerned with a specific information source. As building a domain ontology is a challenging task sufficient tool support has to be provided to build that ontology. A growing number of ontology editors exist [Duineveld et al., 1999]. The choice of a tool has to be based on the special needs of the domain to be modeled and the knowledge of the expert.

Annotation of Information Sources Once a common vocabulary exists, it can be used to annotate different information sources. In this case annotation means that the inherent concept hierarchy of an information source is extracted and each concept is described by necessary and sufficient conditions using the terminology built in step one. The result of this annotation process is an ontology of the information source to be integrated. The annotation will normally be done by the owner of an information source who wants to provide better access to his or her information. In order to enable the information owner to annotate his information he has to know about the right vocabulary to use. It will be beneficial to provide tool support also for this step. We need an annotation tool with different repositories of vocabularies according to different domains of interest.

Semantic Translation of Information Entities The only purpose of the steps described above was to lay a base for the actual translation step. The existence of ontologies for all information sources to be integrated enables the translator to work on these ontologies instead of treating real data. This way of using ontologies as surrogates for information sources has already been investigated in the context of information retrieval [Visser and Stuckenschmidt, 1999]. In that paper we showed that the search for interesting information can be enhanced by ontologies. Concerning semantic translation the use of ontologies as surrogates for information sources enables us to restrict the translation on the transformation of type information attached to an information entity by manipulating concept terms indicating the type of the entity.

The new concept term describing the type of an information entity in the target information source is determined automatically by a inference engine that uses ontologies of source and target structures as classification knowledge. This is possible, because both ontologies are based on the same basic vocabulary that has been built in the first step of the integration approach.

5.2.2 Required Infrastructure

In order to enable a terminological reasoning system to actually relate concepts, besides interoperability of the logical languages involved (compare chapter 4), we also have to make assumptions about the knowledge represented. These assumptions directly refer to the two solutions to the explication dilemma mentioned above, because reasoning across ontologies requires a shared basic vocabulary (reduction to syntax) and the description of concepts in both ontologies in terms of logical expressions over these shared terms (reduction to logic).

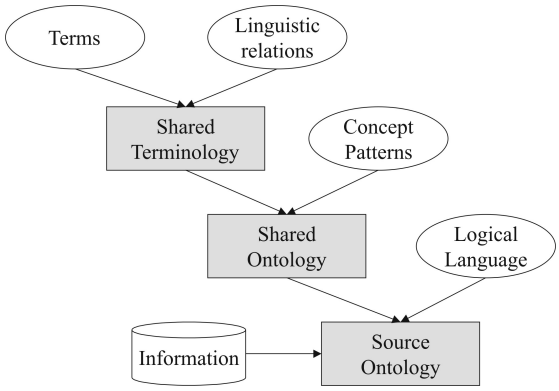


Figure 5.2: Conceptual Architecture of the Translation Knowledge

We distinguish between shared terminology and shared ontology. The shared terminology consists of terms assumed to have a unique meaning across different classifications. These terms are structured by relations borrowed from linguistics, i.e. synonym (equivalent terms), hypernym (broader term) and hyponym (more specialized term) relations. Formally, we define a shared terminology as a set of words and a partial function over pairs of words:

Definition 5.1 (Shared Terminology) *A shared terminology is a tuple* $\langle W, l \rangle$ *where* W *is a set of words and* $l : W \times W \rightarrow \{syno, hyper, hypo\}$ *is a partial function from the set of all pairs of terms into a set of identifiers specifying whether the first term is a synonym, a hypernym or a hyponym of the second.*

This shared terminology is linked to the specific integration problem using *structural patterns*. The structural pattern is a general specification of relations between objects denoted by the words in the shared terminology. In order to be able to apply these relations to information objects, the shared terminology is resembled in a simple logical structure consisting of a set of terms corresponding to words from the shared terminology relations between these terms and a set of axioms. The axioms define the synonym, hypernym and hyponym relations between terms in terms of the subsumption relation as defined on page 4.2.1.

Definition 5.2 (Shared Ontology) *A Shared Ontology is a tuple* $\langle ST, T, R, A\rangle$ *where* $ST = \langle W_L, l_L\rangle$ *is a shared terminology,* T *is a basic set of terms,* R *is a set of relations* $R \subseteq T \times T$ *and* A *is a set of axioms of the form* $T_i \sqsubseteq T_j$ *if the following conditions hold:*

- $T \subseteq W_L$
- *for each pair of words* (W_i, W_j) *holds*
  - *if* $l((W_i, W_j)) = hyper$ *then* $W_j \sqsubseteq W_i$ *is in* A
  - *if* $l((W_i, W_j)) = hypo$ *then* $W_i \sqsubseteq W_j$ *is in* A
  - *if* $l((W_i, W_j)) = syno$ *then* $W_i \sqsubseteq W_j$ *and* $W_j \sqsubseteq W_i$ *are in* A

From the point of expressiveness, this shared ontology is very similar to a model in RDF schema, because it defines a hierarchy of terms (classes in RDF schema) as well as a set of relations (properties) with corresponding range and domain restrictions. This correspondence enables us to use RDF schema in order to encode shared ontologies as a basis for defining information semantics.

Shared ontologies provide us with a vocabulary we can use in order to specify the semantics of information in different sources. This semantics, however, has to be defined individually for different information sources. In order to capture the semantics of types or assessments used in an information source, we need a richer language, because their meaning does almost never directly correspond to a term in the shared ontology. We therefore define a source ontology, an ontology that defines the meaning of specific classifications used in the source, to consist of a set of class definitions. These definitions are legal expressions over terms from the shared ontology build using a terminological language that defines operators for the relations also defined in the shared ontology:

Definition 5.3 (Source Ontology) *A source ontology is a tuple* $\langle S, C, L, d\rangle$ *where* $S = \langle ST_S, T_S, R_S, A_S\rangle$ *is a shared ontology,* C *is a set of class names not from the set of terms in* S*,* L *is a terminological language and* d *is a function that assigns expressions* δ_i *to class names* C_i *in* C *such that:*

- δ_i *is a legal expression in* L
- *the operators in* L *only refer to relations in* R_S*.*
- L *is defined over* T_S

In the following we refer to δ_i *as the definition of* C_i *which is denoted by* $d(C_i)$*.*

Given a source ontology we can perform terminological reasoning over the definition of classes contained therein by considering the set of axioms from the shared ontology, the definitions of relations and the set of class definitions. Together, these elements form a terminological knowledge-base that can be used by suitable description logic reasoners in order to provide standard inference services such as classification and retrieval. How these inference services are used for retrieval and integration will be discussed in chapter 7.

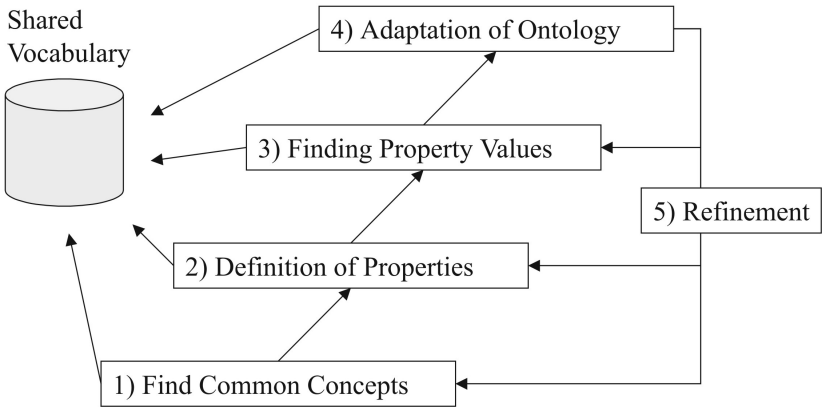


Figure 5.3: Steps of the Development Process

5.2.3 Building the Infrastructure

The integration process sketched above relies on the existence of a shared ontology suitable to define concepts from all terminologies to be integrated in sufficient detail. This requirement is a challenge with respect to ontology building. In order to support this difficult task, we propose a development strategy that is tailored to the purpose of building shared ontologies. In this section we give an overview of the development process.

The Process

The proposed strategy is based on stepwise-refinement. It consists of five steps that are executed in an iterative process resulting in a partial specification of the shared ontology. The last step of each run is an evaluation step that triggers one of the previous steps in order to extend and refine the ontology if necessary.

Figure 5.3 illustrates the process model, the individual steps are briefly described below.

Step 1: Finding Common Concepts The first step is to examine the translation task. Asking the question "what do I want to translate?" leads to a concept that subsumes all classes from the source and destination systems. Because this concept makes a semantic translation from one source into another possible we call it bridge concept. While defining its properties and attribute values we achieve the needed shared vocabulary. The most general bridge concept is "top", a concept that subsumes every other possible concept. For an exact classification it is recommended to choose the bridge concept as concrete as possible. If needed, more than one bridge concept can be defined to enable semantic translation.

Step 2: Definition of Properties The next step is to define properties that describe the chosen bridge concepts. A car, for instance, can be described through its color, its brand, its price, etc.

Step 3: Finding property values Once we have defined the properties, we search for values which can fill the attributes. These "fillers" are the main part of the shared vocabulary.

Step 4: Adapt ontology The use of existing sources of information will not always be sufficient to describe all concepts of an information source. We sometimes have to handle very specific distinctions made in a system that hardly occur in standard terminologies. In order to capture these subtle differences we have to invent application specific terms as part of the shared vocabulary.

Step 5: Refine Definitions The introduced strategy follows the "evolving" life cycle. It allows the engineer to step back all the time to modify, add and remove ontology definitions, e.g. refining the bridge concept or integrate further taxonomies into the shared vocabulary.

Each of the steps modifies a different aspect of the shared ontology. While step 1 is concerned with the central concept definition, step 2 defines slots, step 3 integrates existing taxonomies, and step 4 generates application-specific taxonomies. This fact is useful in order to determine where to go back to if the evaluation step reveals the inability to describe a certain aspect of a terminology to be integrated.

Sources of Information

The use of the ontology to be built as a common basis for communication between systems makes it necessary to stay as closely as possible to a vocabulary and conceptualization of the domain that is widely accepted as a standard. In order to meet this requirement, we use several sources of information to build upon. These information sources are existing ontologies and thesauri as well as scientific classifications and data catalogues.

Top-Level Ontologies are mainly used to find the bridge concept which acts as a template for the definition of all terms to be translated. In most cases, the bridge concept is obvious, however, the use of an upper level ontology provides us with a vocabulary which is partly standardized.

Scientific Classifications are another form of standards describing the conceptualization of a domain. Classifications like taxonomies of animals or plants are common knowledge which can be used to specify concepts from domain-specific ontologies.

Domain Thesauri contain typical terms used in an application domain, therefore they are a natural source for finding concept names for the shared ontology. Further, many thesauri contain at least free-text definitions of the terms included. These definitions provide guidance for the definition of concepts.

Linguistic Thesauri are used to supplement information taken from domain-specific thesauri. In contrast to the specialized vocabulary defined in domain-specific thesauri, linguistic thesauri can be used to identify correspondences between terms found in different information sources. Especially, we use linguistic thesauri to expand the search for definitions of terms to their synonyms.

Data Catalogues finally contain the definitions of the terminology to be modeled. Therefore they define the concepts to be modeled and are the basis for evaluating the expressiveness of the shared ontology at a specific point in the modeling process.

In the course of the modeling process, we stick as closely as possible to the information from the sources mentioned above. Therefore, the selection of these sources, though not discussed in this paper is already an important step when building a shared ontology.

5.3 Applying the Approach

We performed a case study in order to assess the general strategy described above. In the following we will describe the task of this case study and give an impression of how the strategy helps to build the models needed to solve it.

5.3.1 The Task to be Solved

Geographical information systems normally distinguish different types of spatial objects. Different standards exist specifying these object types. These standards are also called catalogues. Since there is more than one standard, these catalogues compete with each other. To date, no satisfactory solution has been found to integrate these catalogues. In our evaluation we concentrate on different types of areas distinguished by the type of use.

In order to address the semantic translation problem we assume a scenario where the existing land administration database that is normally based on the ATKIS catalogue, which is the official standard for most administration should be updated with new information extracted from satellite images of some area. Satellite images are normally analyzed using image processing techniques resulting in a segmentation of different areas which are classified according to the CORINE landcover nomenclature, a standard for the segmentation and classification of satellite images. The process of updating the land administration system with this new data faces two main problems:

1. The boundaries of the objects in the database might differ from the boundaries determined on the satellite image.

2. The class information attached to areas on the satellite images and the type information in the land administration system do not match.

The first problem is clearly out of the scope of our investigation, but the second one is a perfect example of a semantic translation problem. A successful integration of the two information sources will come with the following benefits for the user of the systems: (a) *integrated views* and (b) *verification*. An integrated view from the users perspective merges the data between the catalogues. This process can be seen as two layers which lay on top of each other. The second option gives users the opportunity to verify ATKIS-OK-250 data with CORINE land cover data or vice versa.

The basis for our experiment is a small CORINE landcover data-set containing information about the town 'Bad Nenndorf' in Lower Saxony. This data-set is available from the German Environmental Agency in different formats and classifications and can therefore be used to compare and evaluate results. In our case study, we want to find out, whether land-use classes from the corine-landcover data-set can be semantically translated into the classification used by the ATKIS catalogue. Such a translation could be the basis for both, the generation of an integrated view on the information in both systems and for a validation of ATKIS data with up-to-date satellite images.

5.3.2 The Information Sources

The ATKIS catalogue [AdV, 1998] is an official information system in Germany. It is a project of the head surveying offices of all the German states. The working group offers digital landscape models with different scales from 1:25.000 up to 1:1.000.000 with a detailed documentation in corresponding object catalogues. We use the large scale catalogue OK-1000. This catalogue offers several types of objects including definitions of different types of areas. Figure 5.5 shows the different types of areas defined in the catalogue.

CORINE landcover [European Environmental Agency, 1999a] is a deliverable of the CORINE programme the European Commission carried out from

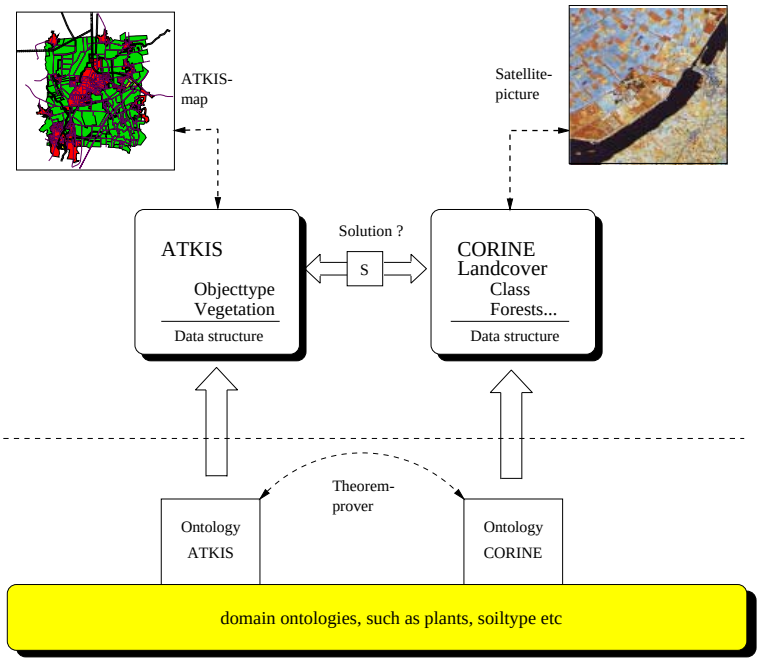


Figure 5.4: Deductive Integration of Geographic Information

1985 to 1990. The results are essentially of three types, corresponding to the three aims of the Programme: (a) an information system on the state of the environment in the European Community has been created (the CORINE system). It is composed of a series of data bases describing the environment in the European Community, as well as of data bases with background information. (b) Nomenclatures and methodologies were developed for carrying out the programs, which are now used as the reference at the Community level. (c) A systematic effort was made to concert activities with all the bodies involved in the production of environmental information especially at international level. The nomenclature developed in the CORINE programme can be seen as another catalogue, because it also defines a taxonomy of area types (see figure 5.6) with a description of characteristic properties of the different land types.

The task of this example is that the data of CORINE database has to be converted in the ATKIS database. Of course, this transformation can be viewed as a special case of an integration task demonstrating all the problems which can occur. Besides the obvious structural heterogeneity problems, the main problem relies on the reconciliation of the semantic heterogeneity caused by the use of different classification schemes.

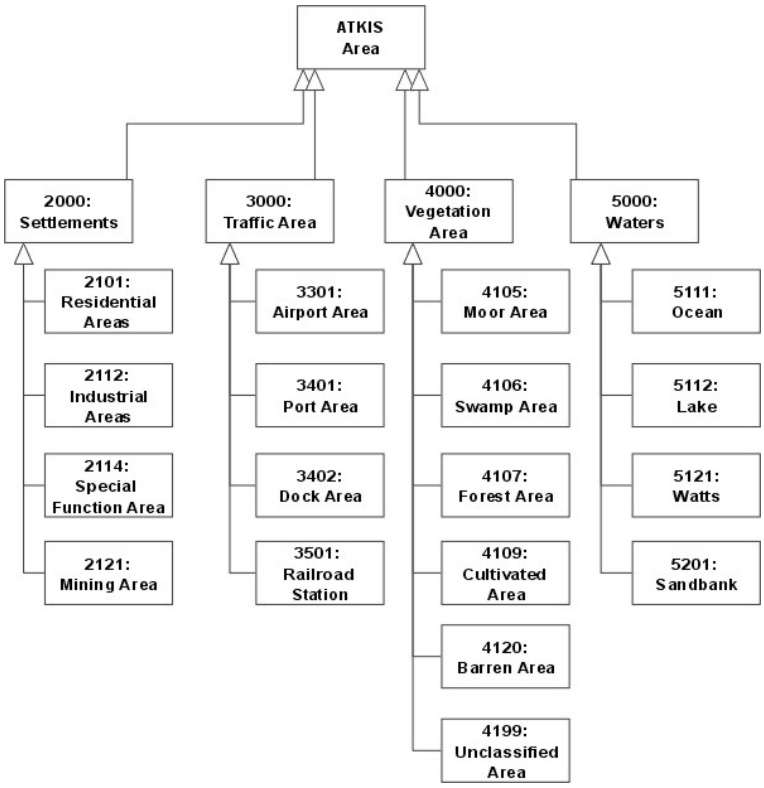


Figure 5.5: Taxonomy of land-use types in the ATKIS-OK-1000 catalogue

The classification schemes of land-use types in figures 5.5 and 5.6 illustrate this problem. The set of land types chosen for these catalogues are biased by their intended use: while the ATKIS catalogue is used to administrate human activities and their impact on land use in terms of buildings and other installations, the focus of the CORINE catalogues is on the state of the environment in terms of vegetation forms. Consequently, the ATKIS catalogue contains fine-grained distinctions between different types of areas used for human activities (i.e. different types of areas used for traffic and transportation) while natural areas are only distinguished very roughly. The CORINE taxonomy on the other hand contains many different kinds of natural areas (i.e. different types of cultivated areas) which are not further distinguished in the ATKIS catalogue. On the other hand, areas used for commerce and traffic are summarized in one type.

Despite these differences in the conception of the catalogues the definition of the land-use types can be reduced to some fundamental properties. We identified six properties used to define the classes in the two catalogues. Beside *size* and *general type of use* (e.g. production, transportation or cultivation) the

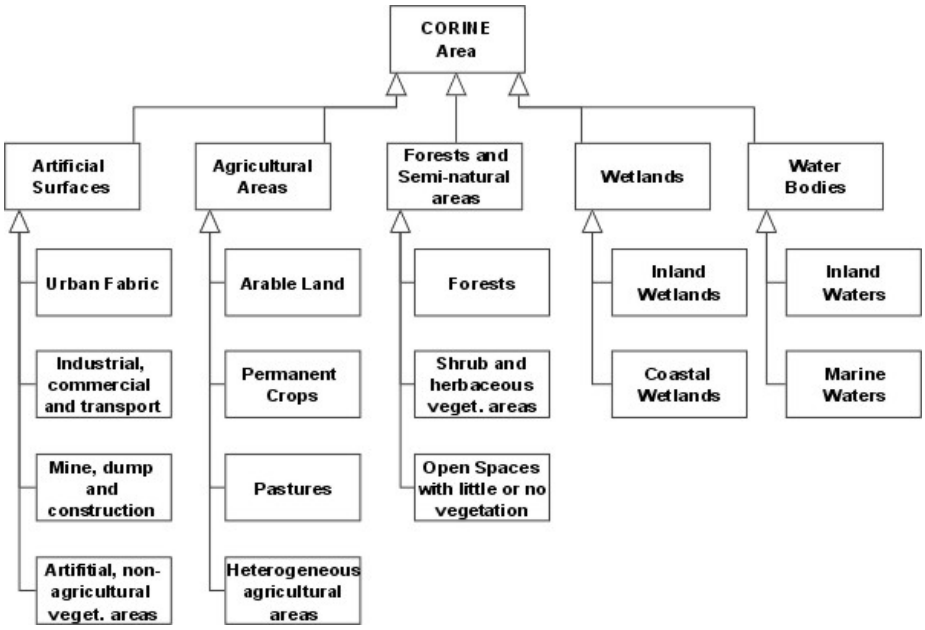


Figure 5.6: Taxonomy of land-use types in the CORINE land cover nomenclature

kinds of structures built on top of an area, the *shape of the ground* and *natural vegetation* as well as kinds of *cultivated plants* are discriminating characteristics.

5.3.3 Sources of Knowledge

For this specific integration task we chose several sources of information to be used for guiding the development process. We briefly describe these sources in the following.

UpperCyc Ontology

The UpperCyc, developed by the CyCorp corporation [Lenat, 1995] (http://www.cyc.com), is an upper-level ontology that captures approximately 3,000 terms of the most general concepts of human consensus reality. There is also a full Cyc knowledge base (KB) including a vast structure of more specific concepts descending below the UpperCyc, the so called top-level ontology. It contains millions of logical axioms – rules and other assertions – which specify constraints on the individual objects and classes found in the real world. Therefore the Upper Cyc ontology provides a sufficient common grounding for applications.

GEMET

The "GEneral Multilingual Environmental Thesaurus (GEMET)" [European Environmental Agency, 1999b] is a polyhierarchically structured thesaurus which covers approximately 5.400 terms and their definitions organized by groups, themes, and terms. GEMET has been created by merging different national and international thesauri. Analysis and evaluation work of numerous international experts and organizations let to a core terminology of generalized environmental terms and definitions. GEMET ensures validated indexing and cataloguing of environmental information all over Europe. Where available, synonyms or alternate terms can be found likewise.

WordNet

WordNet [Fellbaum, 1998], developed by the Cognitive Science Laboratory at Princeton University, is an on-line lexical reference system whose design is inspired by current psycholinguistic theories of human lexical memory. English nouns, verbs, adjectives, and adverbs are organized into synonym sets, each representing one underlying lexical concept. Different relations link the synonym sets.

Standard Taxonomies

Scientific taxonomies can be found in many sources, like books or the internet. For this example we looked into the Google Webdirectory (http://directory.google.com/Top/Science/Biology/Flora_and_Fauna) to obtain a classification of plant life. It is in no circumstances complete, but it satisfies our needs in this case study.

5.4 An Example Walkthrough

Based on the information described above we built up a first version of a shared ontology which should be used to solve the integration task mentioned in the last section. In this section we sketch the first development cycle of this ontology using the concrete modeling activities to illustrate the different steps of our strategy.

Step 1: Finding Bridge Concepts

Looking at the given example scenario as described in chapter 5.2.3 it is quite obvious to choose a concept like "area" or "region", because all land-use classes are some kind of special "regions", or in other words, "region" subsums all land-use classes. We search for the term "region" in the "Upper-CYC" and get the following definition:

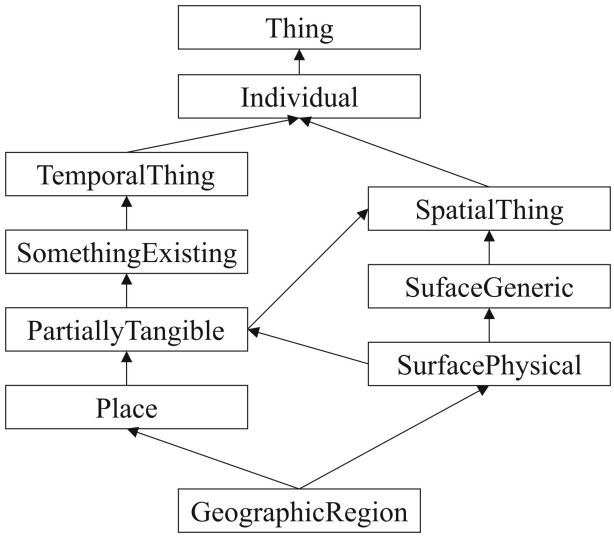


Figure 5.7: Geographical Region in the Upper-CYC

> *GeographicalRegion: A collection of spatial regions that include some piece of the surface of PlanetEarth. Each element of GeographicalRegion is a PartiallyTangible entity that may be represented on a map of the Earth. This includes both purely topographical regions like mountains and underwater spaces, and those defined by demographics, e.g., countries and cities* $[\cdots]$ *"*

Figure 5.7 shows the hierarchical classification of the concept in the Upper-CYC. The definition fits very well, so finally we choose "Geographical Region" as our bridge concept. For further refinement we write it down in the OIL notation.

```
Class-def Geographical-Region
```

Step 2: Definition of Properties

Now we have to find possible attributes for the bridge concept. We look for "Geographical Region" in the GEMET, but the search does not give any results. In that case the decomposition of the search phrase may give better results. For "Geography" and "Region" we get this definitions out of GEMET:

> *Geography: "The study of the natural features of the earth's surface, comprising topography, climate, soil, vegetation, etc. and man's response to them."*

> *Region: "A designated area or an administrative division of a city, county or larger geographical territory that is formulated according to some biological, political, economic or demographic criteria."*

Here, some attributes are clearly recognizable. For example "vegetation" is a biological criterion that defines a region, and it is also part of the scientific field geography. We update the bridge concept by defining a slot "vegetation" and adding it to the bridge concept.

```
Slot-def vegetation
    Domain Geographical-Region

Class-def Geographical-Region
```

Step 3: Integration of Standard Taxonomies

To get possible "attribute values" or "filler" for the slot "vegetation", we take another look into GEMET. Vegetation is defined as:

> *"The plants of an area considered in general or as communities [· · ·]; the total plant cover in a particular area or on the Earth as a whole."*

We also check the synonym "flora", found in WordNet:

> *"The plant life characterizing a specific geographic region or environment."*

The attribute "vegetation", respectively "flora", can be filled with terms out of plant life like "tree" or "rose" for instance. A good top concept is "plants", because many scientific taxonomies of plants exists. The Swedish botanist Carlous Linaeus established 1753 a classification of plants. His work is considered the foundation of modern botanical nomenclature. In the Google Webdirectory we can access the plant kingdom with more than 10000 entries online. We integrate this taxonomy into our vocabulary.

Now it is possible to describe classes from the land-use catalogues. The term "coniferous forest" in the CORINE context is defined as:

> *"Vegetation formation composed principally of trees, including shrub and bush understories, where coniferous species predominate."*

In our vocabulary we find the term "Coniferophyta", comprising the conifers, which are trees or shrubs that bear their seeds in cones, without the protection of a fruit like angiosperms. This leads to the following OIL class:

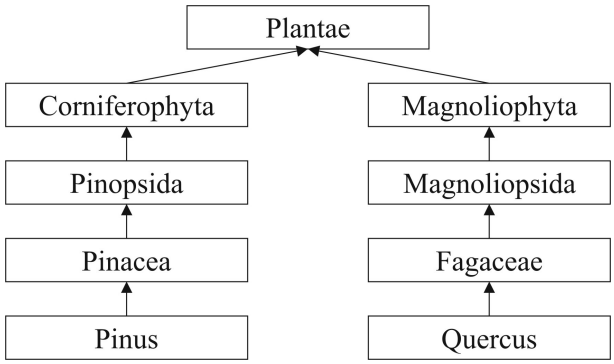


Figure 5.8: Extract from scientific plant taxonomy

```
class-def  Coniferous-Forest
  subclass-of  Geographical-region
  slot-constraint vegetation value-type Coniferophyta
```

The Division Magnoliophyta of the plant kingdom consists of those organisms commonly called the flowering plants, or angiosperms. The flowering plants are the source of all agricultural crops, cereal grains and grasses, garden and roadside weeds, familiar broad-leaved shrubs and trees, and most ornamentals. So, it is easy to describe the next CORINE class "broad-leaved forest":

```
class-def  Broad-leaved_Forest
  subclass-of  Geographical-region
  slot-constraint vegetation value-type Magnoliophyta
```

A "mixed forest" in the CORINE nomenclature consists of conifers and broad-leaved trees.

```
class-def  Mixed_Forest
  subclass-of  Geographical-region
     slot-constraint vegetation has-value Magnoliophyta
     slot-constraint vegetation has-value Coniferophyta
```

Step 4: Adapt vocabulary

A closer look at the definition of the CORINE forest classes reveals that the classes are defined through the existence of trees and shrubs. Just using the term "Magnoliophyta" does not prevent the classification of a region covered with orchids as a broad-leaved forest (Orchidaceae is a subclass of Magnoliophyta). The mentioned taxonomy classifies plants according to their way of re-production therefore distinguishing angiosperm and gymnosperm trees, shrubs, and flowers. To handle this problem we need a more general distinction.

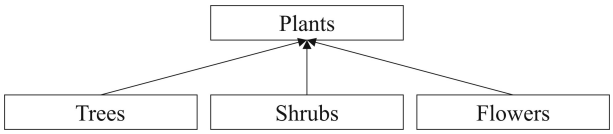


Figure 5.9: Supplementary Plant Classification

Figure 5.9 shows a simple extension of the vocabulary that enables a more robust definition of the CORINE forest classes.

```
class-def  Coniferous_Forest
  subclass-of  Geographical-region
  slot-constraint vegetation value-type Coniferophyta and (trees or shrubs)

class-def  Broad-leaved_Forest
  subclass-of  Geographical-region
  slot-constraint vegetation value-type Magnoliophyta and (trees or shrubs)

class-def  Mixed_Forest
  subclass-of  Geographical-region
  slot-constraint vegetation
    has-value Coniferophyta and (trees or shrubs)
    has-value Magnoliophyta and (trees or shrubs)
```

The shared vocabulary developed so far allows us to specify many different vegetation areas found in the land-use catalogues:

```
class-def  Pastures
  subclass-of  Geographical-region
  slot-constraint vegetation value-type Poaceae

class-def  vineyards
  subclass-of  Geographical-region
  slot-constraint vegetation value-type Vitis

class-def  Rice_fields
  subclass-of  Geographical-region
  slot-constraint vegetation value-type Oryza
```

Step 5: Evaluation and Revision

Not all CORINE landcover classes can be described after this first process cycle. "Mineral extraction sites", for instance, are defined as:

> *"Areas with open-pit extraction of minerals (sandpits, quarries) or other minerals (opencast mines). Includes flooded gravel pits, except for river-bed extraction."*

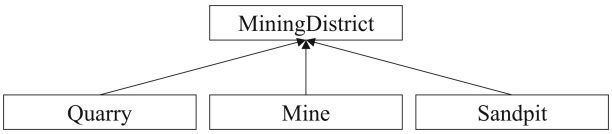


Figure 5.10: Mining sites from the GEMET thesaurus

No vegetation is mentioned, so the bridge concept must be refined. We go back to step 2 "defining properties" and search for another attribute. The definitions of "region" and "geography" show some anthropological aspects, like "man's response" or economic criteria. So we define a new slot "anthroposphere" and add it to our bridge concept:

```
slot-def anthroposphere
    Domain geographical-region

slot-def vegetation
    Domain geographical-region

class-def Geographical-Region
```

In the topic area 'anthroposhere' of the Gemet thesaurus we find the term "mining district", a district where mineral exploitation is performed. We integrate the partial taxonomy into the vocabulary (figure 5.10).

This special vocabulary can be used to simulate one-to-one mappings by using equality axioms. The CORINE class "mineral extraction sites" could be described as followed.

```
class-def  Mineral-extraction-sites
  subclass-of  Geographical-region
  slot-constraint anthroposphere has-value mining-district
```

In a similar way, we proceed iterating the process cycle until all terms from the two catalogue systems can be modeled as a specialization of the bridge concept. A further advantage of this strategy is the fact that the same process will be employed when additional terminologies are to be integrated as well. We cannot guarantee that the shared ontology also covers a new terminology, but our strategy already provides guidance for the adaption of the ontology.

Lessons Learned

The examples given above already show that the method leads to better results than an early hands-on approach described in [Stuckenschmidt et al., 2000]. In this early case study, we developed the shared vocabulary solely by relying on textual description of the two catalogues mentioned above. The development

strategy proposed here results in a shared model that uses mostly standardized terms and is well integrated with existing higher level ontologies.

We also managed to describe more concepts with less properties. The use of the vegetation property for example turned out to be sufficient for describing about half of all concepts from both information sources. We explain this with the richer vocabularies for describing different vegetation types we got from scientific classifications.

An interesting side-effect of the more controlled development is a harmonization of the structure of logical expressions used to define concepts. We explain this by the fact that the strategy forces us not to describe a concept completely without comparing it to other definitions. The strategy rather forces us to define restrictions for a particular property for many concepts in parallel. This direct comparison makes it easier to capture the specific structure of the logical expression required in contrast to the definition of other concepts.

Conclusions

While the question of how to represent ontologies is a prerequisite for information sharing, the central question is how to actually capture information semantics in ontologies. A strategy is needed that determines what kinds of ontologies are needed and how they can be build. This strategy has to trade-off globalized representations that provide a common basis for defining and comparing information semantics and local representations that capture the specific conceptual choices made in the design of individual information sources. In order to be comparable, these local definitions should be based on terms defined globally. Linguistic resources and top-level ontologies provide guidance in the choice for a global vocabulary. The representational framework defined in the first part of this thesis then provides operators for composing these basic terms to more complex concept definitions and to perform terminological reasoning.

In real applications the most important question is often not how to arrange ontologies, but how to actually build these ontologies. This problem has been widely recognized and some methodologies have been developed to support the development of ontologies. In most cases, these methodologies are very general and do only provide basic guidance for the development of an ontology infrastructure. In our approach the notion of a shared vocabulary is essential and the development of this vocabulary therefore deserves special attention. We had good experiences with a strategy that follows a bottom-up approach that takes the actual integration problem as a starting point and consults general models like top level ontologies and linguistic resources only if necessary. The resulting vocabularies are general enough to cover at least a certain class of integration problems. We think that this is more valuable than a general top-down ap-

proach because it solves real world problems without losing the connection to basic ontological principles.

Chapter 6

Metadata Generation and Management

In the previous chapter, we defined a general architecture for describing information semantics in terms of ontologies that are derived from shared terminologies of a domain and encoded using terminological languages in order to give them a clean, model-theoretic semantics. We also presented a strategy for building these ontologies. What is still missing at the moment is a strategy of how to actually relate information to its semantics encoded in source ontologies. In this chapter, we will discuss how weakly structured information can be linked to the ontology infrastructure described in the previous chapters using metadata. Further, we discuss intelligent ways of maintaining this metadata. Our goal is to provide tools and methods for the generation and the management of such metadata, because the effort of relating huge amounts of information to ontology may become a serious hazard to successful information sharing. We illustrate the developed methods in a case study we carried out using an existing Web-based information system and enriching it with semantic information.

Acknowledgement A shorter description of metadata management with the WebMaster Workbench was published as [Stuckenschmidt and van Harmelen, 2001a]. Adapting the approach to generate metadata models based on a source ontology was proposed in [Stuckenschmidt and van Harmelen, 2001b]. Both are co-authored by Frank van Harmelen. The idea of automatically learning classification rules used by the WebMaster Workbench have been reported in [Stuckenschmidt et al., 2002] with was co-authored by Jens Hartmann and Frank van Harmelen. A description of the conducted experiments is contained in [Hartmann and Stuckenschmidt, 2002] which is co-authored by Jens Hartmann.

Kashyap and Sheth [Kashyap and Sheth, 1998] analyze so-called global information systems where many different and possibly heterogeneous information repositories have to be integrated. In order to achieve interoperability between these repositories they propose to link the information repositories to ontological terms using metadata (compare figure 6.1).

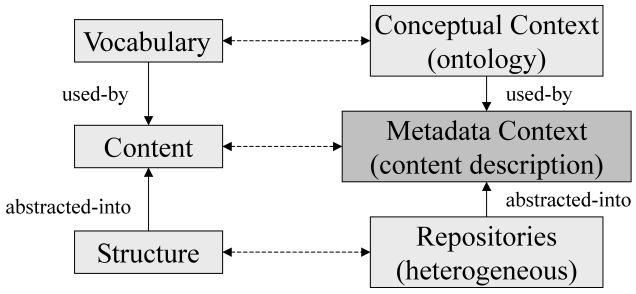


Figure 6.1: Assigning Semantics to Information [Kashyap and Sheth, 1998]

We refer to this view on global information systems, because it describes a way of deploying the source ontologies described in the last chapter on the Web. The notion of a source ontology developed in the last chapter directly corresponds to the conceptual context described by Kashyap and Sheth, because source ontologies define the meaning of terms used in an information repository. Following Kashyap and Sheth we have to define a metadata context for information repositories that uses terms from the source ontologies in order to give an abstract description of the information contained in an information repository.

In this chapter, we will develop a strategy for assigning ontological terms to information items and resources using metadata models. We start recapitulating the role of metadata for information sharing and identify critical problems. We then present an integrated approach for generating and managing metadata that is based on source ontologies.

6.1 The Role of Metadata

A common approach to the problem of information heterogeneity is to provide so-called metadata, i.e. data about the actual information. As this description is very general and different scientific communities use very different notions of metadata, we first have to define our view on metadata more clearly. For this purpose we use the following distinctions made by [Kashyap and Sheth, 1998]:

Content-Independent Metadata is data about information that does not

directly relate to the content of the information it describes. It rather describes the context and the environment the information is created and maintained in. Content independent metadata includes the author of a document or the date of its creation.

Content-Dependent Metadata is data about information that is derived from the information, but it does not describe the information content, but rather additional property that directly follow from the content. Examples for content-dependent metadata are the size of a document, the number of words or pages or the language a document is written in.

Content-Based Metadata directly reflects the content of an information source, but it adds information or structure that helps to process the original information more efficiently. Examples for content-based metadata are document vectors or full-text indices.

Content-Descriptive Metadata finally is data about information that provides an abstract description of the content of an information resource. It helps to summarize the content of an information source and judge whether it is suitable for a particular purpose. Examples of content descriptive metadata are keyword lists, glossaries or assignments to thematic categories.

The different kinds of metadata cover many aspects of information: technical data about storage facilities and access methods co-exist with content descriptions and information about intended uses, suitability and data quality. Concerning the problem of finding and accessing information, the role of metadata is two-fold: On the side of information providers it serves as a means of organizing, maintaining and cataloguing data, on the side of the information users meta-data helps to find, access and interpret information. We briefly discuss this two-fold view in the following paragraphs.

6.1.1 Use of Meta-Data

Organizing large information repositories is a difficult problem. While standard databases provide sophisticated technology for data organization and maintenance, heterogeneous repositories likes data warehouses, federated databases, and especially the World Wide Web suffer from the problem of heterogeneity that demands for sophisticated organization methods. Concerning this problem, metadata can serve different purposes:

- **Structuring:** Meta-Data can be used to structure heterogeneous information by specifying topic areas, keywords and relations to other information. This kind of meta information can be used to organize information along different dimension like topic, date, author, etc.

- **Maintenance:** Meta-data can help to maintain data by providing information about authors, date of creation and expiry of the information. This information helps to locate outdated information or to find the person who is responsible for changes.
- **Cataloguing:** The bigger an information repository becomes the more important it is to have an overview of the information that is actually present. This can be done by creating information systems based on metadata cataloguing the information available.

Similar problems can be identified in connection with the use of information on the World Wide Web. Standard databases are mostly homogeneous systems with well-defined query languages that can be used to access information available in the database. On the Web, a user first of all has to find the information needed, before it can be used. Then the information may be present in different kinds of data formats and structures. Last but not least, information that seems to fit a users need can be tailored for a completely different purpose and can therefore be hard to use. Again, metadata can be used to tackle these problems:

- **Search:** By providing topic areas, keywords and content summaries, as well as information about intended use, metadata can be used in order to identify information sources on the Web without having to search every single Web page. Being confronted with the rapidly growing size of the Internet, this ability can be predicted to be very important in the near future.
- **Access:** Meta-data related to technical properties of an information source like format, encoding, links to tools or wrappers can significantly reduce the effort required to process available information.
- **Interpretation:** Using information does not only require that information can be accessed, the data also has to be interpreted by the remote system. Information about the terminology used, assumptions made and knowledge required to interpret the content can help both human users and intelligent systems to really understand the content of an information source.

We conclude that the use of metadata is important in order to support the handling and the use of information in heterogeneous environments like the World Wide Web because metadata helps to organize large information repositories and access these repositories efficiently.

6.1.2 Problems with Metadata Management

The considerations made above clarify the need for metadata especially for Web-based information systems. Therefore it is not surprising that various

approaches for modeling and using metadata have been developed. Standards evolved that cover different aspects of metadata, especially the syntax for coding, the model structure and content of a metadata model. Some of these standards are:

- **Syntactic Standards:** Features for encoding metadata are already included in common HTML [Ragget et al., 1999]. Meta-tags can be used in order to specify attributes and corresponding values of an HTML document. Recently, RDF [Champin, 2000] has been proposed as an XML application especially designed for the representation of meta-information about information on the World Wide Web. However, these approaches only define the encoding syntax in order to enable Web-browser to operate on the metadata.
- **Structural Standards:** In order to support the development of useful metadata models, a standardization of model structures is an important topic. Structural standards have been defined on top of existing syntactic standards. RDF schema [Brickley et al., 1998], for example, define a model structures similar to frame-based knowledge-representation systems. Topic Maps [Pepper and Moore, 2001] are another important important approach prescribing representation elements to describe information about the contents of information resources.
- **Content Standards:** While approaches like RDF schema or topic maps define structural elements for representing metadata, there is still no guidance with respect to the kind of data to be stored about information in order to organize and use information efficiently. As a consequence, content standards for metadata have been proposed. One of the most important content standards is the so-called Dublin Core [Weibel, 1999] that defines a set of metadata elements for diocuments. These elements can be encoded using different syntactic standards, e.g. HTML and RDF).

The standards mentioned above provide good guidance to design and encode metadata for information resources on the world-wide Web. However, there are still some severe problems that are addressed neither by structural nor by content standards. These problems are concerned with the relation between information and metadata about it. Some of the most important are:

- **Completeness:** In order to provide full access to an information source, it has to be ensured that all the information is annotated with the corresponding metadata. Otherwise, important or useful parts of an information source may be missed by metadata driven search methods or cannot be indexed correctly.
- **Consistency:** Meta-data about the contents of available information is only useful if it correctly describes this contents. In fact, metadata that is

not consistent with the actual information is an even bigger problem than missing metadata, because mechanisms relying on metadata will produce wrong results without warnings.

- **Accessability:** In order to be useful, metadata has to be accessible not only by the information provider but especially for users that want to access it. Therefore, an important question is how a comprehensive description of an information source can be provided and accessed by potential users.

As metadata plays an important role in information sharing by enabling remote programs to find, access and interpret information, we have to provide solutions for the problems mentioned in order to be able to support information sharing. When trying to provide partial solutions for these problems we restrict ourselves to content descriptive metadata, because this type of metadata and especially the use of topic categories provide a very good basis for connecting information with semantic descriptions. We will discuss the process of establishing this connection in the remainder of this chapter.

6.2 The WebMaster Approach

In this section, we present an approach for intelligent metadata management that partially solves the problems mentioned above. The starting point for our presentation is BUISY an existing Web-based information system that serves as an example for the use and problems of metadata on the Web. We will briefly describe this system and the role metadata plays in it. We further present the WebMaster Workbench, a system for the knowledge-based verification of Web sites and show how it can be applied to the BUISY system solving some of the problems mentioned. The results of this application will be the basis for extensions of the WebMaster approach that will be presented in the next section.

6.2.1 BUISY: A Web-Based Environmental Information System

The advent of Web-based information systems came with an attractive solution to the problem of providing integrated access to environmental information according to the duties and needs of modern environmental protection. Many information systems were set up either on the Internet in order to provide access to environmental information for everybody, or in intranets to support monitoring, assessment and exchange of information within an organization. One of the most recent developments in Germany is BUISY, an environmental information system for the city of Bremen that has been developed by the Center for Computing Technologies of the University of Bremen in cooperation with the public authorities. The development of the system was aimed at providing unified access to the information existing in the different organizational units for in-

ternal use as well as for the publication of approved information on the Internet.

Figure 6.2: The Meta-Data Driven Document Search Facility

Metadata plays an important role in the BUISY system. It controls the access to individual Web pages. Each page in the BUISY system holds a set of metadata annotations reflecting its contents and status [Voegele et al., 2000]. The current version of BUISY supports a set of meta tags annotating information about the data-object's type, author, dates of creation and expiration as well as relevant keywords and the topic area of the page. The "Status" meta-tag indicates whether the data-object is part of the Internet or the Intranet section of BUISY.

```
<meta name="Status" content="Freigegeben"/>
<meta name="Typ" content="Publikation"/>
<meta name="Author" content="TJV"/>
<meta name="Date" content="10-04-1999"/>
<meta name="Expires" content="31-12-2010"/>
<meta name="Keywords" content="Wasser, Gewssergte, Algen"/>
<meta name="Bereich" content="Wasser"/>
```

At the moment, this metadata is used to provide an intelligent search facility for publications of the administration concerned with environmental protection. The user selects a document type and a topic area. Based on the input, a list of available publications is generated (see figure 6.2).

6.2.2 The WebMaster Workbench

We have developed an approach to solve the problems of completeness, consistency, and accessibility of metadata identified above. This is done on the basis of rules which must hold for the information found in the Web site, both the actual information and the metadata (and possibly their relationship) [van Harmelen and van der Meer, 1999]. This means that besides providing Web site contents and metadata, an information provider also defines classification rules (also called: integrity constraints) for this information. An inference engine then applies these integrity constraints to identify the places in the Web site which violate these constraints. This approach has been implemented in the WebMaster content management tool, developed by the Dutch company AIdministrator (www.aidministrator.nl). In this section, we will describe the different steps of using the WebMaster Workbench.

Step 1. Constructing a Web site ontology

The first step in the approach to content-based verification and visualization of Web pages is to define an ontology of the contents of the Web site. Such an ontology identifies classes of objects on our Web site, and defines subclass relationships between these classes. For example, pages can be about water. These can again be subdivided into new subclasses: *Gewaesser* (watercourses), *Weser* (a river in Bremen) *Grundwasser* (groundwater) *Abwasser* (wastewater) and *Anlagen* (technical installations). Further, we included some classes corresponding to types of documents that might appear in the system. We chose *Berichte* (reports) and *Verordnungen* (legislations). This leads to a hierarchy of pages that is based on page-contents, such as the example shown in Figure 6.3.

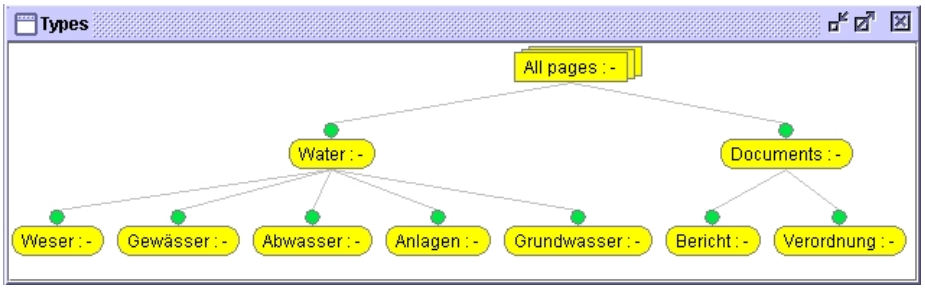


Figure 6.3: An Example Classification Tree

A subtle point to emphasize is that the objects in this ontology are *objects in the Web site*, and not objects in the real-world which are described by the Web site. For example, the elements in the class "river-drainage" are not

(denotations of) different river-drainage systems in the environment of Bremen, but they are *Web pages* (in this case: Web pages talking about river-drainage systems). As a result, any properties we can validate for these objects are properties of the *pages on the Web site*, as desired for our validation purposes.

Step 2. Defining the classification criteria for the ontology

The first step only defines the classes of our ontology, but does not tell us which instances belong to which class. In the second step, the user defines rules determining which Web pages will be members of which class. In this section, we will briefly illustrate these rules by means of three examples.

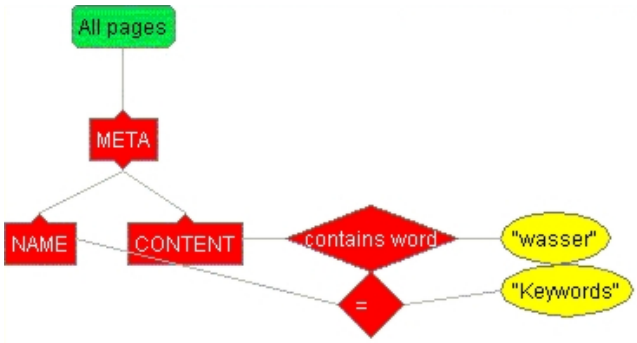


Figure 6.4: Example of a Classification Rule Using Meta-Data

Figure 6.4 specifies that a Web page belongs to the class "water" if the keyword "Wasser" appears in the meta-information of the page. The rule succeeds for example when the following code appears in the Web page:

```
<meta name="Keywords" content="Wasser">
```

In the typical case, a page belongs to a class if the rule defined for that class succeeds for the page. However, it is also possible to define classes by negation: a page belongs to a class when the corresponding rule fails on that page. This is indicated by a rectangle in the class-hierarchy (instead of a rounded box).

Step 3. Classifying individual pages

Whereas the human user of the WebMaster Workbench performs the previous steps, the next step is automatic. The definition of the hierarchy in step 1 and the rules in step 2 allows an inference engine to automatically classify each page in the class hierarchy. Notice that classes may overlap (a single page may

belong to multiple classes). The rule format (adopted from [Rousset, 1997]) has been defined in such a way as to provide sufficient expressive power while still making it possible to perform such classification inference on large numbers of pages (many thousands in human-acceptable response time). After these three steps, we have a class hierarchy that is populated with all the pages of a given site.

6.2.3 Applying WebMaster to the BUISY System

The ability of the WebMaster Workbench to classify Web pages according to the metadata contained in every page enables us to use the system to perform the tasks we claimed to be necessary for metadata management on the Internet, i.e. the validation, aggregation and visualization of the metadata annotations in the BUISY system. At that time the BUISY system contained approximately 1500 pages which are not maintained centrally, but the different topic areas of the systems had been supplemented by different persons after the initial development phase that ended in 1998. Due to this fact, we expected to be faced with incomplete and inconsistent metadata annotations in the different parts of the system. We performed some validation and some aggregation experiments on this metadata which are reported in the next sections.

Validating Metadata

Checking Meta-Attributes and Values After extracting the pages that are actually supposed to contain information, we can start to check the completeness of the annotated metadata. In our analysis, we focused on the meta-information assigning a page to a certain topic area. In the BUISY system this information is stored in the meta-attribute named 'Bereich'. So the first task is to check whether all pages which passed the pre-selection contain the meta-attribute Bereich. The result of this test was rather negative. We found that about one hundred of the six hundred fifty contents pages do not contain the Bereich attribute. Another three pages did contain the attribute but without a value. It is very likely that not all pages which were included into the BUISY system are annotated yet. However, using the WebMaster Workbench, we are able to find these pages and to decide whether metadata has to be added or not.

Check for Missing Keywords The validation of the keyword annotations actually existing in the system is the next step of our analysis. In order to judge the quality of the present annotations we defined some keywords covering important aspects of the information found in the system. We chose the keyword according to the classes described in step 1. We used the keywords to compare the keyword annotations with the contents of the page using a full

text search on the whole page.

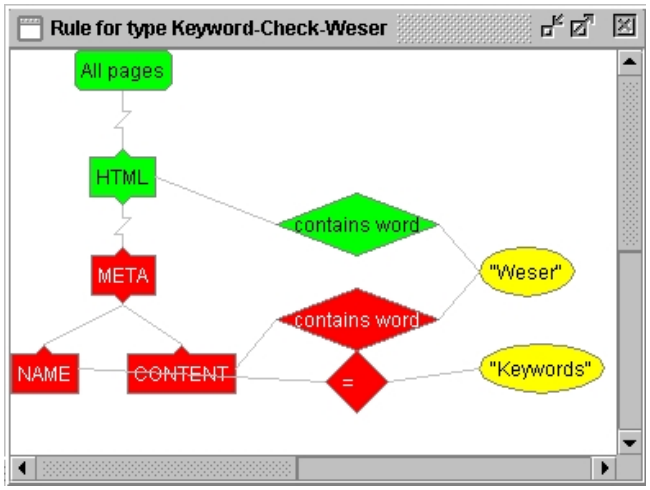


Figure 6.5: Classification Rule for the Detection of Missing Keywords

Figure 6.5 shows a corresponding class definition rule. The rule states that if the Web page contains the word Weser (the main river in Bremen) then there has to be a meta tag where the value of the NAME attribute equals Keywords and the value of the CONTENT attribute contains the word Weser.

The validation revealed that most pages containing a keyword in the text did not have this keyword in the metadata annotation. Using the WebMaster Workbench, we were able to identify these pages and present them to the systems administrator who has to decide if the keyword has to be added.

Aggregating Metadata

The validation of metadata discussed in the previous section is all done on the <META>-tags which are distributed across the 1500 pages of the BUISY system. At construction time, such a distributed organization of the metadata is rather attractive: each page can be maintained separately, containing its own metadata. The authors of pages can directly update the metadata annotations when updating a page, and no access to a central metadata repository is needed. However, when we want to use the metadata to create content-based navigation maps (as in the next section), or as the basis for a metadata-based search engine, such a distributed organization of the metadata is no longer attractive. We would prefer having fast access to a central metadata repository instead of having to make remote access to 1500 separate pages when looking for certain metadata.

Using the validation process described in section 5.2.3 we analyzed the Web site with respect to membership of pages to different topic areas. The result of this step is a classification of pages into a number of classes, based on the application of the classification rules to the <META>-tags in the pages. This yields a populated class-hierarchy of pages. Such a populated class hierarchy can be stored in a combined RDF and RDF Schema format. The following statements are taken from the RDF Schema encoding of the WebMaster type hierarchy. The first three show how the types "water", "Gewaesser" and "Weser" and their subtype relationship are encoded in standard RDF Schema.

```
<rdfs:Class rdf:ID="water"/>

<rdfs:Class rdf:ID="Gewaesser">
   <rdfs:subClassOf rdf:resource="#water"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Weser">
   <rdfs:subClassOf rdf:resource="#water"/>
</rdfs:Class>

...
```

The following is an example of an RDF encoding of instance information: the page at the URL mentioned in the "about" attribute is declared to be a member of the class "water" (and consequently of all its supertypes, by virtue of the RDF Schema semantics.

```
<rdf:Description
    about="http://www.umwelt.bremen.de/buisy/scripts/buisy.asp?
        doc=Badegewaesserguete+Bremen">
   <rdf:type resource="#Gewaesser"/>
</rdf:Description> ...
```

These automatically generated annotations constitute an aggregated description of a Web site that can be used to get an overview of its contents. The annotations are machine-readable, but they are hard to use by a human Web-master. This is the reason why we do not only generate an aggregated metadata model, but also provide a condensed visualization on the basis of the aggregated model.

Metadata Visualization

WebMaster supports the automatic generation of so-called cluster maps about a Web site. A cluster map visualizes an instantiated hierarchy of pages by

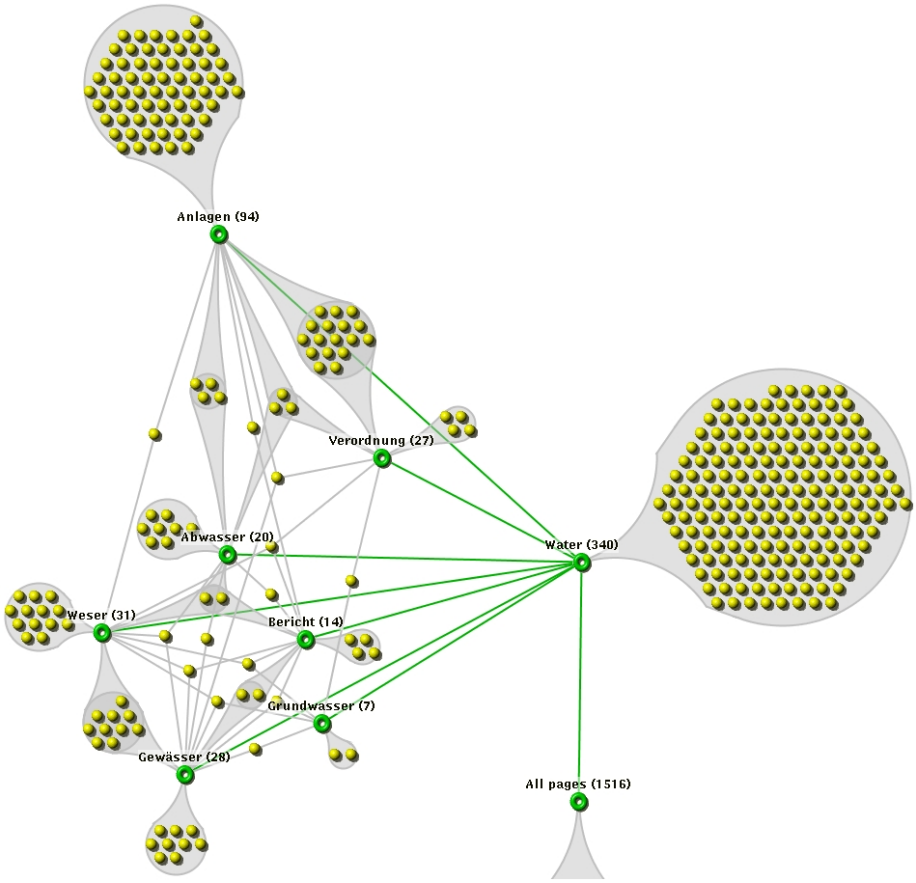


Figure 6.6: Cluster Map of the Water Subsystem

grouping pages from the same class into a cluster. These clusters may overlap if pages belong to more than one class.

The map generated from the classes described above (figure 6.6) shows some interesting features. The first thing that attracts attention is the fact that again most of the pages could not be classified into one of the keyword classes. The better part of the approximately one thousand pages analyzed do not even contain information about the topic area water. This can be explained by the fact that a content map always contains all pages of a Web site. However, there are also many pages which contain relevant contents, but do not belong to one of the keyword classes (page cluster at the right-hand side of the page). The interesting part of the content map is its left side where the pages from the

different keyword classes and their membership in these classes are displayed. We can clearly identify pages about technical facilities and waste water as well as pages containing information about legislation concerning one or both of these topics.

Automatically constructed figures such as figure 7 are compact enough to display many hundreds of pages in a single small image (the map contains 340 pages). This should be compared with the output from traditional search engines, where a set of more than 300 pages is typically presented as 15 pages with 25 URLs each. The format of figure 6.6 is much more usable in practice.

6.3 Learning Classification Rules

We conducted experiments in using the WebMaster Workbench to classify the Web pages in the BUISY system. In the course of this case study eight groups of AI students with some experience in knowledge representation and knowledge-based systems independently specified classification rules reflecting the conceptual structure of the system. This corresponds to the second step in the process described above. It turned out that the creation of structural classification rules is still a difficult and error-prone task. We identified the following reasons for the difficulties:

1. people are not familiar with the structure of the pages
2. people are not familiar with the semantics of logical rules

We propose to address these problems directly by:

1. learn classification rules from examples
2. use a learning approach that is based on logic

In the following we describe an approach of using inductive logic programming (ILP) in order to learn structural classification rules in the spirit of WebMaster. The reason for using ILP instead of more widely used attribute-based learning approaches is two-fold. First of all, the expressiveness of attribute-based learners is mostly restricted to propositional logic while WebMaster rules are first order. In especially, it is often the case that a rule contains a number of binary predicates connected by common variables. In order to be able to learn a large class of rules, we cannot rely on a propositional learner. Secondly, ILP allows us to incorporate complex background knowledge in the learning process, which is not possible with attribute-based learners. In the current state of the work that is reported in the following sections, we did not make much use of background knowledge, however, we want to keep that option for future work. In the remainder of this section, we give a short introduction to ILP. We relate the general definition to our application and present some interesting results that have been achieved. A more detailed description of the approach can be found in [Hartmann, 2002].

6.3.1 Inductive Logic Programming

Inductive Logic Programming (ILP) [Muggleton, 1999] is a techniques that combines inductive learning, the generation and validation of hypotheses from observations with logic programming. The latter provides us with the possibility to learn more complex logical formulae and to use complex background knowledge for guiding the search for hypotheses. We will see later that these possibilities are important with respect to learning structural classification rules. The general inductive logic programming problem can be defined as follows [Muggleton and de Raedt, 1994]:

Definition 6.1 (Normal Semantics of ILP) *Let B, E and H be logical theories where B is given background knowledge and $E = E^+ \wedge E^-$ is given evidence divided into positive and negative evidence. The aim of ILP is to find a hypotheses H, such that the following conditions hold:*

Prior Satisfiability: $B \wedge E^- \not\models \bot$

Prior Necessity: $B \not\models E^+$

Posterior Satisfiability: $B \wedge H \wedge E^- \not\models \bot$

Posterior Sufficiency: $B \wedge H \models E^+$

This definition ensures that the learned hypothesis is complete in the sense that it explains all positive evidence (posterior sufficiency) and that it is consistent with negative evidence and background knowledge (posterior satisfiability). Further, the definition excludes trivial results by claiming that the positive examples are not already explained by the background knowledge (prior necessity) and that background knowledge is consistent with the negative results, because this would enable a logical reasoner to deduce any fact in the theory (prior consistency). Existing approaches implement this general framework in different ways. They can be distinguished by the logical language used to encode evidence, hypotheses and background knowledge and by the learning strategy.

Representation Language: Most ILP approaches use logic programs to describe evidence, hypotheses and background knowledge. While background knowledge and hypotheses can be an arbitrary logic program, evidence is normally given in terms of ground facts. ILP Systems can be separated between *single-predicate* and *multiple-predicate* learners. All evidence given to a single-predicate learner are instances of a single predicate. Instead of a multiple-predicate learner whose examples are instances of more than one predicate.

We used the PROGOL system [Muggleton, 1995]. PROGOL is a multi-predicate learner that uses the logical programming languages PROLOG for

representing knowledge. Any PROLOG programme can serve as background knowledge. Evidence can be encoded in terms of two sets of facts (positive and negative evidence). The hypotheses generated by PROGOL are horn-clauses defining single goal predicate.

Learning Strategy: ILP systems can use different strategies to search the hypothesis space. The main operations to generate such a hypothesis are *specialization* and *generalization* which is called a *top-down* respectively a *bottom-up* strategy. Another distinction can be made on how the examples are given to the learner. Given all examples at once, it is called *batch learning*. At *incremental learning* an example is given one by one and the theory covers the actual set of examples. If there is a possibly interaction between the teacher and the learner while learning is in progress, it is called an *interactive learner* otherwise it is a *none-interactive learner*.

PROGOL is a top-down, non-interactive batch learner. It uses inverse entailment to generate only the most specific hypothesis. An A^* like algorithm is used to search through the hypothesis space. To restrict the hypothesis space (*bias*), the teacher defines first-oder expressions called *mode declarations*. These model declarations restrict the combination of predicates that are considered in the process of finding hypotheses. We will discuss the use these declarations in the next section in more detail.

6.3.2 Applying Inductive Logic Programming

In order to apply inductive logic programming to the problem of learning structural classification rules, we have to answer several questions. These include the representation of the problem in the framework of ILP, the generation of this representation from a given and the task of relating the results produced by the induction step to classification rules used in WebMaster. We will discuss these issues in the following sections.

Problem Representation

In order to use the PROGOL system for generating hypotheses about classification rules for Web pages, we have to encode knowledge about Web pages and their internal structure in PROLOG. For this purpose, we developed a representation scheme consisting of the following set of pre-defined predicates:

- `document(object)`: the constant 'object' represents a document
- `url(object, ADRESS)`: the document represented by 'object' has the URL 'ADDRESS'
- `relation(doc1, doc2)`: there is a directed link between the document 'doc1' and 'doc2'

- `structure(object, CLASS)`: the constant 'object' represents an element tag of type 'CLASS'
- `contains(doc, object)`: the document contains the tag 'object' as a top level element.
- `attribute(parent, object)` the element tag 'parent' contains the attribute 'object'
- `contains(parent, object)` the element 'parent' contains the element 'object' as a child element
- `value(object, 'VALUE')`: 'object' is an element or attribute and it has the value 'VALUE'
- `text_value(object, 'TEXT')`: 'object' is an element or attribute and it has the text 'TEXT'

The distinction between values and text values is necessary, because free text content of texts is normalized and broken up into single words. A text_value predicate is created for every word in a free text passage.

The predicates form the building blocks for representing background knowledge, hypotheses and evidence. Positive and negative evidence is provided in terms of two sets of `document` predicates indicating that certain constants represent documents of a certain class or not. The PROLOG representation of the structure of all of these documents is provided as background knowledge. Then system then creates hypotheses in terms of rules that have the `document` predicate on the left-hand side and a conjunction of the predicates described above on the right-hand side.

Generating the Representation

In order to be able to use an ILP learner for the acquisition of syntactic classification rules, the structure of the documents serving as positive and negative examples have to be translated into the representation described above. We assume that documents are present in XML or XHTML format. Unfortunately, most of the documents came in less standardized form, partly containing syntactic errors. Therefore all training examples were semi-automatically cleaned and tidied up. We used HTML Tidy[1] and its Java version JTidy[2] for this task. In cases where heavy syntactic errors were found by the software we fixed them manually.

The next step to obtain a usable training set is the *syntactical translation* of the training examples. A Web document like a HTML or an XML Document contains predefined tags which describes structure (in particular relations

[1] http://www.w3.org/People/Raggett/tidy/
[2] http://lempinen.net/sami/jtidy/

inside a document or between other documents) and layout of documents. The complete translation process is described here in a very abstract way: (i) Every document is parsed into a DOM tree. We used Apache JXERCES 2.0 for this task. (ii) Our written Java program then walks through the DOM tree. Depending on a predefined translation scheme all desired tags are translated into PROLOG clauses. (iii) The positive and negative examples are written into a file which represents the training set. (iv) In order to enable the system to perform a restricted kind of learning on the text of a page, simple normalization techniques are applied that convert the words of a text into lower case letters, removes special symbols as well as words from a stop list and inserts a list of the remaining words in the PROLOG notation.

Relation to WebMaster Rules

Rules used in WebMaster have the following general logical form for our rules and constraints also known as *positive database update constraints* in the database area. A rule for characterizing the class C has the following structure:

$$C \leftarrow \left(\forall \vec{x} [\exists \vec{y} \bigwedge_i P_i(x_k, y_l)] \implies [\exists \vec{z} \bigwedge_j Q_j(x_k, z_m)] \right) \tag{6.1}$$

where the $\vec{x}$, $\vec{y}$ and $\vec{z}$ are sets of variables, and each of the P_i and Q_j are binary predicates. Furthermore, the classification rules are always *hierarchical* (i.e. recursion free), and also free from negation. This class of formulae is less expressive than full first order logic over the predicates P_i and Q_j (because of the limited nesting of the quantifiers), but is more expressive than Horn Logic (because of the existential quantifier in the right-hand side of the implication).

If we restrict this general rule format, and we drop the existential quantification in the right-hand side, we are left with a very restricted form of logic programs: *hierarchical normal programs over binary predicates.*

This correspondence to logic programs can be used to translate hypotheses generated by PROGOL into the WebMaster rule format. Looking at the general rule format we see that PROLOG clauses are a special case of the rules where all predicates P_i are assumed to be true and omitted. Therefore a WebMaster rule generated by PROGOL has the following format:

$$C \leftarrow \left(\forall \vec{x}, [\exists \vec{z} \bigwedge_j Q_j(x_k, z_m)] \right) \tag{6.2}$$

In this case the predicates Q_j describe necessary structures for pages of class C. These predicates are taken from the set of predicates described above.

6.3.3 Learning Experiments

Our aim is to identify obvious structural regularities within classes of Web pages. The PROGOL system allows us to use background knowledge for focusing the learning process on different kinds of such regularities that are likely to discriminate between classes of Web pages. These regularities have to be specified in terms of goal predicates that can be specified by a horn clause.

In order to assess the quality of our learning approach, we determine the accuracy of the learned rules in terms of the ratio of correctly classified pages. We use the following notation to refer to classification results:

P(A): correct positive (pages from the class covered by the rule)

$\neg P(A)$: false negative (pages from the class not covered by the rule)

$\neg P(\neg A)$: correct negative (pages not from the class that are not covered by the rule)

$P(\neg A)$: false positive (pages not from the class that are covered by the rule)

Using these definitions, we use the following definition of accuracy:

$$Accuracy = \frac{P(A) + \neg P(\neg A)}{P(A) + \neg P(A) + \neg P(\neg A) + P(\neg A)} * 100 \tag{6.3}$$

The accuracy is determined by splitting the set of all Web pages into a training-set and a test-set, where 70% of all pages belong to the training and 30% to the test set. Further, we used a ratio of 1:2 between positive and negatives examples (for each positive example there are two negative ones) Below we give accuracy measures for our experiments based on this ratio.

Tested Criteria

As mentioned above, the PROGOL system allows to focus the search for hypotheses on parts of the overall knowledge available. This possibility, implemented in so-called *mode-declarations* can be used to define classification criteria to be used in generated classification rules. In our case, we can use mode declarations to prescribe what kinds of Web page structures should be tested by the system. For this purpose, we first invent the new predicate *descendant* as a transitive closure of the *contains* predicate mentioned above.

$$descendant(A, B) \leftarrow contains(A, B)$$
$$descendant(A, C) \leftarrow contains(A, B) \wedge descendant(B, C)$$

A mode declaration is a definition of a new predicate that should be used on the right hand side of classification rules. In order to enable the learner to

generate hypotheses containing this new predicate it has to be related to the knowledge provided by the reasoner. We do this by defining new predicates in terms of the basic set of predicates used to describe page structures. In the following, we briefly present the mode declarations we used for the generation of structural classification rules.

Document Titles: Document titles often contain information about the type of the page. Personal home pages for example will in most cases contain the word 'homepage' in its title. We therefore use the predicate *doctitle* that relates a page to word occurring in its title as a first criterion. The corresponding predicate is defined as follows:

$$doctitle(D, T_i) \leftarrow descendant(D, Q) \wedge structure(Q, 'title') \wedge contains(Q, W) \wedge text\_value(W, T_i)$$

In the following we refer to this mode declaration as H1.

Meta-Data Existing metadata annotations on a Web page are an obvious choice for defining classification criteria. We use HTML meta-tags as a search criterion. As different Web sites may use different metadata attributes, we do not further restrict the search to specific metadata such as keywords or authors. The corresponding predicate that locates HTML metadata on a Web page in a general way is the following:

$$metatag(I, N, C) \leftarrow descendant(D, I) \wedge structure(I, meta) \wedge attribute(I, Q) \wedge attribute(I, W) \wedge structure(Q, x) \wedge value(Q, N) \wedge structure(W, y) \wedge value(W, C)$$

In the following we refer to this mode declaration as H2.

E-mail Addresses More complex Web site often contain links to a special contact e-mail. Assuming that different persons are responsible for different topic areas according to their field of expertise we can exploit the occurrence of a certain mail address on a page for defining classification criteria. The corresponding predicate is defined as follows:

$$mail(D, Q, W) \leftarrow descendant(D, S) \wedge structure(S, 'a') \wedge attribute(S, R) \wedge structure(R, 'href') \wedge value(R, 'mailto:' + Q + '@' + W)$$

In the following we refer to this mode declaration as H3.

Links Web sites are often organized in a hierarchical way. Different topic areas are often accessed via a top level page containing a table of content or an introductory page. In order to exploit these common access points for classification, we use links to other pages as another classification criterion. These links are identified by an anchor tag with an *href* attribute. The corresponding predicate is the following:

$$relation(D_1, D_2) \leftarrow descendant(D_1, Q) \wedge structure(Q, a) \wedge attribute(Q, W) \wedge structure(W, href) \wedge value(W, Z) \wedge url(D_2, Z)$$

In the following we refer to this mode declaration as H4.

An Example of Using Mode Declarations

We illustrate the impact of the search guidance using results we achieved on classifying the Web site of the University of Bremen. the goal was to learn classification rules that uniquely identify pages of the research group on theoretical computer science. For this purpose we used about 150 pages of that group as positive and about 300 other pages from the university Web site as negative examples. Table 6.3.3 shows generated rules for the different mode declarations and the accuracy of the rules.

Experiment A1-0		TrainingSet0
	TZI - Theorie	
Mode Dec.	*Hypotheses*	*Acc.*
H 1	document(A) :- doctitle(A,research).	100
H 2	document(A) :- metatag(A,keywords, theoretical).	100
H 3	document(A) :- relation(A,B), relation(B,C), mail(C,helga,'informatik.uni-bremen.de').	86,82
H 4	document(A) :- relation(A,B), url(B,'[URL]/cs/ref.num.html'). document(A) :- relation(A,B), relation(A,C), url(B,'[URL]/projects.html'). document(A) :- relation(A,B), relation(A,C), url(B,'[URL]/cs/ref.num.html').	86,82
URL: `http://www.tzi.de/theorie`		

Table 6.1: An Example of the criteria

The results show the different kinds of classification rules we get when using different mode declarations for guiding the search process. Using the page title as a criterion, we find out that the pages of the theoretical computer science group are exactly those that contain the word 'research' in their title. An analysis of metadata shows that the keyword 'theoretical' uniquely identifies

the pages we are interested in. We get more surprising results that still have an accuracy of more than 85% when analyzing e-mail addresses and links to other pages. For the case of e-mail addresses we find out that most pages are linked over steps with a page that contains the mail address of the secretary of the group. If we only consider links, we see that most pages are linked to pages containing references and to a page listing projects of the group.

Summary of Results

We conducted an experiments in learning classification rules assigning the Web pages of the Web based environmental information systems of the following cities or federal states, respectively:

- Bremen: `www.umwelt.bremen.de`
- Vienna: `www.ubavie.gv.at`
- Bavaria: `www.umweltministerium.bayern.de`

We applied our learning approach in order to sort pages in the se systems into the topic areas waste, soil, air, nature and water. In the following, we present the results we achieved for the BUISY system (see section 6.2.1). The complete experiment including the two other systems is reported in [Hartmann and Stuckenschmidt, 2002].

BUISY Summary					
Class	*P(A)*	$\neg P(A)$	$\neg P(\neg A)$	$P(\neg A)$	*Acc.*
Abfall	13	0	26	0	100
Boden	11	0	22	0	100
Luft	28	0	56	0	100
Natur	20	1	42	0	98,41
Wasser	58	0	116	0	100

Table 6.2: Summary of the Learning Results for the BUISY Systems

Table 6.2 shows that we achieved an accuracy of almost 100%. The reason for this is the existence of metadata annotations on almost all pages that directly link the pages to the topic areas. The real benefit of the learning approach, however, is its ability to find classification criteria that are not obvious. In order to discover such unexpected patterns as well, we defined a learning strategy on top of the PROGOL system. Once a valid hypothesis is found, it is stored in a separate solution file. Then all occurrences of the defining structures are deleted from the training data and the learner is run on the data again. This process is repeated until no more valid hypotheses are found. As a result of this strategy we get alternative definitions of the different classes. Some of the more interesting results are discussed in the next section.

6.3.4 Extracted Classification Rules

Beside the rather trivial results we achieved in classifying metadata based on pre-existing classifications encoded in meta-tags, we were also able to extract some more surprising classification rules. This section gives an overview of the rules with more than 50% accuracy that have been found by the learner.

Class 'Abfall' The first class of Web pages concerned with waste management. We used a sample of 45 positive and 90 negative examples. The resulting rules for the different mode declarations are shown in table 6.3.

Experiment B1-0		TrainingSet0
	BUISY - Abfall	
Mode Dec.	*Hypothese(n)*	*Acc.*
H 1	document(A) :- doctitle(A,abf).	74,36
H 2	document(A) :- metatag(A,bereich,abfall).	100
H 3	-	-
H 4	-	-
H 1-4	document(A) :- metatag(A,bereich,abfall).	100

Table 6.3: Experiment B1-0: BUISY - Abfall

We can see that the main classification criterion was the pre-defined classification encoded in the 'Bereich' attribute. Beside this about 75% of all pages of this class had the acronym 'abf' in their title.

Class 'Boden' The second class of pages is concerned with soil protection. For the generation of classification rules for this class we used a sample of 37 positive and 74 negative examples. Table 6.4 shows the results of the learning process.

The results for this class are similar to the one before. Beside the pre-defined classification, only the analysis of the page title produced results. This time, the learner found more that one word frequently occurring page titles. The combination of these word leads to a higher classification accuracy of about 90 %.

Class 'Luft' The third class considered contains pages about air pollution. We used a sample of 94 positive and 184 negative examples to learn classification rules for this class of Web pages. The results are shown in table 6.5.

For this class of Web pages we did not only get results on metadata and page titles, but also for relation between documents. It turned out that about 70 % of all pages of this class are linked to a special search page, where users can access air pollution information for their particular area. Other interesting

Experiment B2-0		TrainingSet0
	BUISY - Boden	
MD	*Hypothese(n)*	*Acc.*
H 1	document(A) :- doctitle(A,bodenschutz). document(A) :- doctitle(A, boden). document(A) :- doctitle(A, bo).	90,91
H 2	document(A) :- metatag(A,keywords,bodenschutz). document(A) :- metatag(A,keywords,boden).	100
H 3	-	-
H 4	-	-
H 1-4	document(A) :- metatag(A,keywords,bodenschutz). document(A) :- metatag(A,keywords,boden).	100

Table 6.4: Experiment B2-0: BUISY - Boden

Experiment B3-0		TrainingSet0
	BUISY - Luft	
MD	*Hypothese(n)*	*Acc.*
H 1	document(A) :- doctitle(A,karte). document(A) :- doctitle(A,blues). document(A) :- doctitle(A,ost). document(A) :- doctitle(A,so2diagramm). document(A) :- doctitle(A,lu). document(A) :- doctitle(A,stickstoffoxiddiagramm). document(A) :- doctitle(A,aktuelle). document(A) :- doctitle(A,verkehr). document(A) :- doctitle(A,ozondiagramm). document(A) :- doctitle(A,staubdiagramm). document(A) :- doctitle(A,stickstoffoxid).	95,24
H 2	document(A) :- metatag(A,expires,thu). document(A) :- metatag(A,bereich,luft).	100
H 3	-	-
H 4	document(A) :- relation(A,B), url(B,'[URL]/p_strassensuche').	69,05
H 1-4	document(A) :- metatag(A,expires,thu). document(A) :- metatag(A,bereich,luft).	100
URL: `http://www.bremen.de/Web/owa`		

Table 6.5: Experiment B3-0: BUISY - Luft

points are the fact that the rules learned for page titles associate the pages with words that are not obvious but rather refer to specialized terms from the area.

Class 'Natur' The fourth class consists of pages about the protection of plants and animals. The sample for this class consisted of 71 positive and 142 negative examples. We were able to generate rules based on title and metadata of the pages. The corresponding rules are shown in table 6.6.

Experiment B4-0		TrainingSet0
	BUISY - Natur	
MD	*Hypothese(n)*	*Acc.*
H 1	document(A) :- doctitle(A,naturschutzgebiete). document(A) :- doctitle(A,nsg). document(A) :- doctitle(A,richtlinie). document(A) :- doctitle(A,vogelschutzgehoelz). document(A) :- doctitle(A,naturschutzgebiet). document(A) :- doctitle(A,vogelschutzgebiet).	92,06
H 2	document(A) :- metatag(A,author,'zdl30-13'). document(A) :- metatag(A,author,brendel). document(A) :- metatag(A,bereich,naturschutz).	98,41
H 3	-	-
H 4	-	-
H 1-4	document(A) :- doctitle(A,richtlinie). document(A) :- metatag(A,author,'zdl30-13'). document(A) :- metatag(A,bereich,naturschutz).	98,41

Table 6.6: Experiment B4-0: BUISY - Natur

Beside the page title that again contained some rather specialized terms like 'vogelschutzgebiet', we found rules that linked the pages to a specific author. In this case 'zdl30-13' identifies a special position in the organization of the environmental administration. We can conclude that this position includes the obligation to create and maintain the information on this specific topic area.

Class 'Wasser' The last class considered in our case study is concerned with water pollution. This part is the largest topic area of the BUISY system. We were able to use a sample consisting of 194 positive and 388 negative examples. The results for the class of water protection pages are summarized in table 6.7.

The results for this class also contained some surprises. First of all, most characteristic words in the document titles have no connection at all with the topic of water protection. The combined use of all mode declaration also produced a rule that identifies the pages of this class in terms of the tool used for their creation. In this case, the metatag 'generator' that is automatically added

Experiment B5-0		TrainingSet0,1
	BUISY - Wasser	
MD	*Hypothese(n)*	*Acc.*
H 1	document(A) :- doctitle(A,wa). document(A) :- doctitle(A,landes). document(A) :- doctitle(A,von).	94,25
H 2	document(A) :- metatag(A,bereich,wasser).	100
H 3	-	-
H 4	-	-
H 1-4	document(A) :- metatag(A,bereich,wasser). document(A) :- doctitle(A,fuer), metatag(A,generator,microsoft).	100

Table 6.7: Experiment B5-0: BUISY - Wasser

to Web pages by the tool Microsoft 'Frontpage' is found to be characteristic for the class of pages.

Conclusions

The experiments showed that our learning approach can be successfully applied. It produces very good results on the identification of existing metadata which already helps to classify unknown systems. In the absence of metadata, the approach is able to find various other classification criteria like words occurring in page titles or links to other Web pages. In this case the average accuracy is significantly lower but still we managed to achieve results that are comparable to other work reported in literature. A good example is the dataset used in th WebKB project [Craven et al., 2000]. We reached an accuracy of about 65% by just applying our approach without customizing it to the task.

6.4 Ontology Deployment

In the last section, we described a mostly automated approach for assigning Web pages to classes in a hierarchy. We used the WebMaster Workbench to classify pages based on structural classification rules and showed that these rules can be learned from examples. If we compare this approach to the ontology infrastructure proposed in chapter 5 we face the problem, that the notion of ontology used by WebMaster differs from our view on source ontologies. While WebMaster operates on a very simple form of ontology consisting only of a hierarchy of classes while source ontologies as defined in chapter 5 consist of complex logical definitions. In this section, we show that despite the difference between the notions on ontologies used, we can use the metadata generation approach in order to deploy source ontologies by assigning Web pages to classes in the ontology in a mostly automatic way.

6.4.1 Generating Ontology-Based Metadata

Our approach for connecting information resources with ontologies relies on the definition of a source ontology to consist of a plain class hierarchy that is connected to logical definitions by a corresponding mapping (compare definition 5.3). This definition enables us to use the class hierarchy independently from the logical definition of the classes in the hierarchy and use it in the WebMaster Workbench. Further we can use the mapping to logical definitions to perform reasoning on class hierarchy and instances.

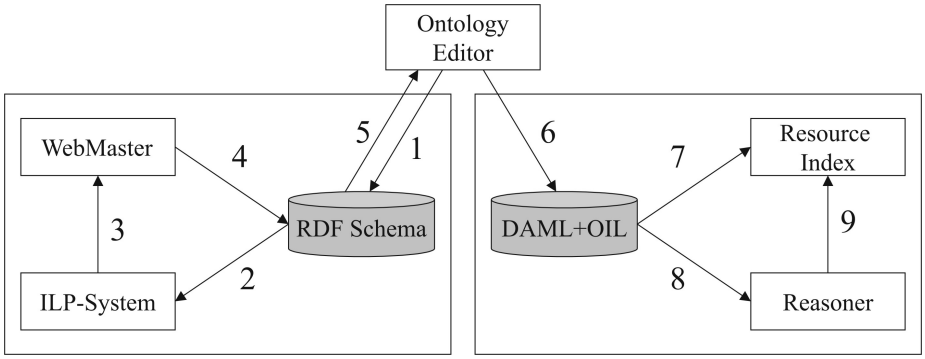


Figure 6.7: Deployment Strategy for Source Ontologies

Figure 6.7 illustrates the process of generating metadata in terms of Web page categorization from pre-existing source ontologies having been built in the way described in chapter 5. Assuming that a complete source ontology exists, the steps of this process are the following:

1. In the first step, we export the class hierarchy of the source ontology as an RDF schema definition as input for the rule learning and Web page categorization process.

2. The exported classification is used for determining goal classes for the ILP system. Based on the class hierarchy, the user determines examples for the learning process.

3. After the generated classification rules have been validated by the user, they are transformed into the WebMaster format and imported into the Workbench.

4. Using the generated classification rules, the WebMaster Workbench assigns the pages of the Web site to classes from the hierarchy and exports the assignment by extending the RDF schema model of the hierarchy.

5. In the ontology editor, the RDF schema model that has been extended by instance information is linked to the complete definition of the ontology that include logical definitions of classes.

6. The complete ontology that now include instance information in terms of classified individuals representing Web pages is exported as a DAML+OIL model for further processing.

7. On the basis of the class names defined in the ontology a Web site index is created in terms of a dynamic data structure that can be queried by other systems.

8. The complete DAML+OIL model is shipped to a description logic reasoner. The reasoner is used to verify the ontology and the instance information. Further implicit subclass and membership relations are derived.

9. The membership relations that have been found by the reasoner (including those already created by WebMaster) are used to insert the information about Web pages in the index structure.

This process can mostly be implemented with existing technologies. Besides PROGOL and WebMaster, we used the OILed Editor [Bechofer et al., 2001] to create source ontologies. The editor supports the export of ontologies in both RDF schema and DAML+OIL. The editor is directly integrated with the description logic reasoner FaCT [Bechhofer et al., 1999] that can be used for perform terminological reasoning on the exported ontology. The use of these reasoning services is described in the next section.

6.4.2 Using Ontology-based Metadata

One of the major benefits of basing metadata on source ontologies is the already mentioned reasoning support for a limited number of tasks concerned with ontology management.

Consistency Checking: The reasoner is able to check the satisfiability of the logical model of the ontology. In particular, inconsistent concept definitions are detected. If we, for example, defined animals to have four legs and we try to include an instance of the class animal with five legs, the reasoner will find the contradiction.

Computation of Subclass Relations: An ontology normally contains two different kinds of sub-class relations: explicitly defined relations from the class hierarchy and implicit subclass relations implied by the logical definitions of concepts. The latter can be detected using reasoner for terminological languages and included into the ontology thus completing it.

Deriving Class Membership: A special case of the computation of subclass relation is the automatic classification of individuals. Terminological languages normally allow us to describe an individual by its relation to other

individuals without naming all classes it belongs to. The reasoner will find the classes we omitted in the definition. An example would be if we had defined our dummy page to be about the 'Sodenmattsee' without assigning it to a special topic area. However, we stated that the domain of the 'about' relation is the class topic area and we defined water-pollution-control to be concerned with watercourses. This information provided and the fact that the 'Sodenmattsee' is a lake and therefore a watercourse enables the reasoner to decide that our dummy page should be classified as belonging to the topic area 'water pollution control'.

Making use of these reasoning services we can check the result of the metadata generation for consistency. This is necessary because the criteria used to describe classes in the systems only refer to syntactic structures of the page contents. Especially, the WebMaster Workbench has no possibility to check whether the classification of a page makes sense from a logical point of view. For example, we can include a description of the administrative units in our ontology and classify pages according to the unit which is concerned with the specific topic of the page. We will define the units to be mutually disjoint because the competency is strictly separated. If we now classify one page to belong to both units we get a clash in the logical model. In this case, we have to check the page and assign the right administrative unit by hand. Thus the logical models helps us to find shortcomings of the generated model.

The second benefit of the logical grounding of the metadata model is the possibility to derive hidden membership relations. This is important because the RDF metadata schema makes some assumptions about implicit knowledge. Examples of these assumptions can be found in [Champin, 2000]. We use the following axiom as an example:

$$\frac{\mathcal{T}(r, \texttt{rdf:type}, c_1) \wedge \mathcal{T}(c_1, \texttt{rdfs:subClassOf}, c_2)}{\mathcal{T}(r, \texttt{rdf:type}, c_2)}$$

The equation states that every resource r (i.e. Web page) that is member of class c_1 (indicated by the triple $\mathcal{T}(r, \texttt{rdf:type}, c_1)$) is also member of class c_2 ($\mathcal{T}(r, \texttt{rdf:type}, c_2)$) if c_1 is a subclass of c_2 ($\mathcal{T}(c_1, \texttt{rdfs:subClassOf}, c_2)$). This correlation can easily be computed using the FaCT reasoner by querying all super-concepts of a given concept. The result of this query can be used to supplement the description of a page. The description of the page referred to above, for example, will be extended with the following statement.

```
... <rdf:type resource="#watercourse"/> ...
```

Using this mechanism, we are able to build a site index that provides efficient access to Web pages by the topic class they belong to.

Conclusions

Metadata plays a central role in information sharing and in information processing in general. It establishes the connection between information sources and ontologies that explicate the meaning of their content. In weakly structured environments this is harder than in structured ones. XML documents, for example, can be directly linked to an ontology on the basis of the tags used in the documents by relating tags to classes or relations in the ontology. In the absence of a real data structure the connection either has to be loose or we have to spend much more effort on the task of establishing the connection.

We claim that the assignment of individual Web pages to classes in an ontology provides a good trade-off between the strength of the connection and the effort of establishing it. We show that Web page classification can be done using classification rules that refer to the structure of HTML documents. The resulting classification can be used for content-based navigation and search. We also demonstrated that structural classification rules can be generated in a mostly automatic way using techniques from machine learning. Though using a very limited learning approach we achieved classification results with an accuracy of ninety percent and more. Our results show that the approach is successful though there is still potential for improving the learning method.

In principle, Web page classification can be done without relating to an ontology, but using a source ontology as a starting point for the classification enables us to benefit from its formal semantics. In especially, we can use terminological reasoning to support the metadata creation process by verifying classification results against the definitions of the classes involved and by driving implicit classifications that are implied by the semantics of the ontology.

Part III

Information Sharing

Chapter 7

Integration and Retrieval

In the last chapter, we discussed how information sources can be semi-automatically enriched by semantic information. We showed how metadata carrying this semantic information can be generated and used in order to provide better access to Web-based information systems. However, all considerations were only focused on a single information system. The idea of information sharing, however , implies that we have different systems that share their information by providing mutual access to information. In this chapter, we show how information can be retrieved and transformed between different system. We show that translations between different ontologies can be approximated using a shared terminology as defined in section 5.2.2. We further describe how this transformation can be exploited for content-based information filtering across different systems.

Acknowledgement: Parts of this work have been published before. The general idea of approximating concepts across ontologies has first been discussed in [Stuckenschmidt and Visser, 2000] which is co-authored by Ubbo Visser. The use of an approximate classifier for information integration and filtering across systems is described in [Stuckenschmidt, 2002b] and [Stuckenschmidt, 2002a].

In order to benefit from the having access to different information systems, we have to provide sophisticated methods to separate relevant from irrelevant information. This problem is also referred to as *information filtering*, which is characterized as the task of removing irrelevant information from an incoming stream of unstructured textual information according to user preferences [Belkin and Croft, 1992]. A different perspective on the same problem is that of information retrieval [Salton and McGill, 1983]. In information retrieval, a collection of information represented by surrogates in terms of content descriptions is searched on the basis of a user query and those documents whose descriptions match the query are returned to the user. Many systems support the *Boolean query model* [Frakes and Baeza-Yates, 1992] that allows

to state queries as Boolean expressions over keywords.

The use of background knowledge has been discussed in classical information retrieval [Yarowsky, 1992, Gaizauskas and Humphreys, 1997] in order to increase the precision and recall of free text queries. Corresponding knowledge models often define relationships between words, such as synonym relation. Using the infrastructure described in chapter 5, we can not only use such relationships that are encoded in the shared terminology (definition 5.1). Our infrastructure transfers these relations into a logical framework, the shared ontology (definition 5.2) that can be used to define specialized terms in different information systems (source ontology, see definition 5.3). This background information is linked to information items by the assignment of Web pages to classes in the source ontology. The assignment to a certain ontological class provides us with a unique interpretation of the meaning of a resource. Using the concepts of a specific ontology, we can state Boolean queries over concept names with maximal precision and recall with respect to the semantics of their definition if all relevant information resources have been assigned to the right ontological categories.

7.1 Ontology-Based Semantics

In [Ciocoiu and Nau, 2000] Ciocoiu and Nau investigated the semantics of theories in declarative languages with respect to a certain ontology that is shared across different logical languages and representations. In this section we briefly review what they call *ontology-based semantics* as a general framework for providing interoperability between ontologies used in different information systems.

7.1.1 The General Idea

The general idea of ontology-based semantics is to restrict the possible interpretations of a set of sentences S_1 represented in a declarative language L_1 from all possible interpretations implied by its axioms to a more concise one that is influenced by a global ontology Ω about the domain of interest. The problem which should be avoided by this restriction is that relations with very different intended meanings might have the same axiomatization. The ancestor relation between persons and the successor relation between natural numbers might serve as an example. The same problem occurs when concepts are atomic, which means that they do not have an axiomatization at all. In this case it is normally not possible to decide (on a purely logical bases) how to translate these concepts or relations.

In order to overcome this problem the authors consider the models of a first-order representation Σ_1 of S_1. These models normally include unwanted models. In order to get rid of these, only those models are considered that are also models

for a shared ontology Ω. In order to make different theories comparable, they are explicitly linked to the shared ontology by rephrasing them using expressions from the ontology. This rephrased theory is called Σ_1^π because it is achieved by a redefinition mapping π. The translatability of one expression into another can now be defined in terms of these interpretations. Ciocoiu and Nau show that S_1 can be translated into S_2 without loosing intended models, if equation 7.1 can be proven by a theorem prover.

$$(\Sigma_1^\pi \cup \Omega) \vdash \Sigma_2^\pi \tag{7.1}$$

The corresponding theorem in fact provides a justification for a deductive translation approaches provided they follow the process outlined in the following section.

7.1.2 Defining Ontology-Based Semantics

The definition of the ontology-based semantics of a set of sentences as defined in [Ciocoiu and Nau, 2000] requires two successive mapping steps. First of all the sentences of the syntactic language have to be transferred into first-order logic ($S \mapsto \Sigma$). This step is called rendering. In the second step which is called interpretation, the sentences are rephrased using expressions from the shared ontology ($\Sigma \mapsto \Sigma^\pi$). We will discuss and exemplify these steps in the following.

Logical Rendering

The first step involved in defining ontology-based semantics is the transformation from the declarative language into an arbitrary first-order language F using a logical rendering function σ. The logical rendering Σ of the set of sentences S in L_1 has the advantage of having the well known semantics of first order logic. Therefore, the logical rendering allows us to apply reasoning methods and provides us with a notion of the logical consequences $Cn(\Sigma)$ of S. In especially $Cn(\Sigma)$ is the first order theory defined by the axioms in Σ. Figure 7.1 illustrates the rendering step.

In the example shown the sentence `(≥ ?x ?y)` is mapped onto the predicate $ge(x, y)$. The sentence `(transitive ≥)` is translated to $(\forall x, y, z)(ge(x, y) \wedge ge(y, z) \implies ge(x, z))$. Further, the assertions `(≥ 3 2)` and `(≥ 2 1)` are mapped on $ge(3, 2)$ and $ge(2, 1)$, respectively. From the semantics of F it follows that $ge(3, 1)$ is a consequence of Σ and therefore contained in $Cn(\Sigma)$.

Interpretation

After rendering the sentences in S they are related to the common ontology Ω. This is done by an interpretation mapping π that basically replaces each n-ary predicate symbol (n-ary functions are considered to be (n+1)-ary predicates) by a formula from the ontology language L_Ω that contains exactly n free

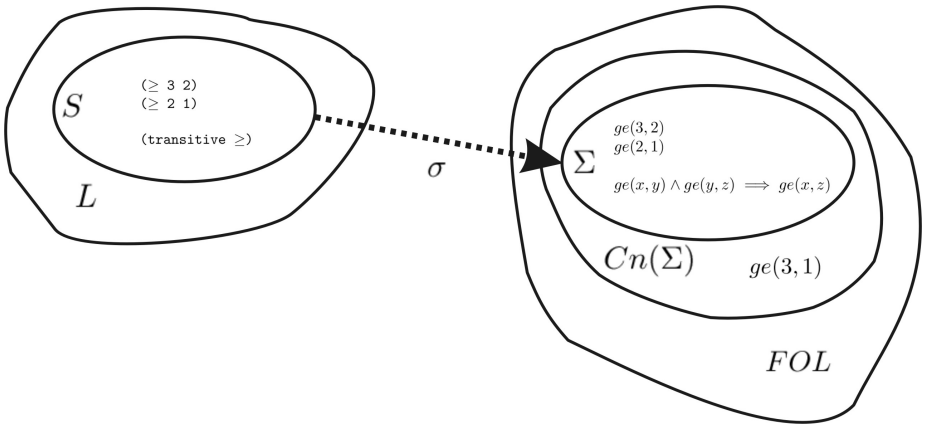


Figure 7.1: Logical rendering of a set of sentences

variables. Formulas from the target theory are now recursively redefined replacing the predicates from F by formulas from the ontology language L_Ω. When mapping universally quantified formulas, a new predicate is introduced that restricts the set of individuals the resulting formula applies to: $\pi((\forall x)\phi) = (\forall x)(\psi(x) \rightarrow \pi(\phi))$. This is necessary, because the models of the target theory may have more individuals than the one of the source theory implying that the translated formula does not hold for all of them, but only for the one associated with the source theory. The resulting interpretation Σ^π together with the axioms of the ontology Ω provides a more precise notion of the intended meaning of the sentences in S. A theory T that includes Σ^π and the ontological axioms defining the meaning of the concepts used therein is called an *interpretation* of S. Figure 7.2 illustrates the interpretation step.

The example in the figure shows how the rendered sentences are interpreted with respect to an ontology specifying the axioms of partial orders with the corresponding axioms (transitivity, reflexivity and anti-symmetry) stated in the language L_Ω. The interpretation π directly translates the two assertions. The rule of transitivity is all quantified, therefore its translation is guarded by introducing the new predicate *number* that restricts the all quantification to elements from the models of Σ. The theory T is an interpretation of Σ^π because it supplements it by the complete axiomatization of the partial order.

7.2 Semantic Integration

In this section we adopt the notion of translatability introduced by Ciocoiu and Nau for the problem of handling multiple classification systems. Analyzing the requirements for performing translations, we will see that the approach for explicating information semantics described in section 5.2.2 largely fulfills

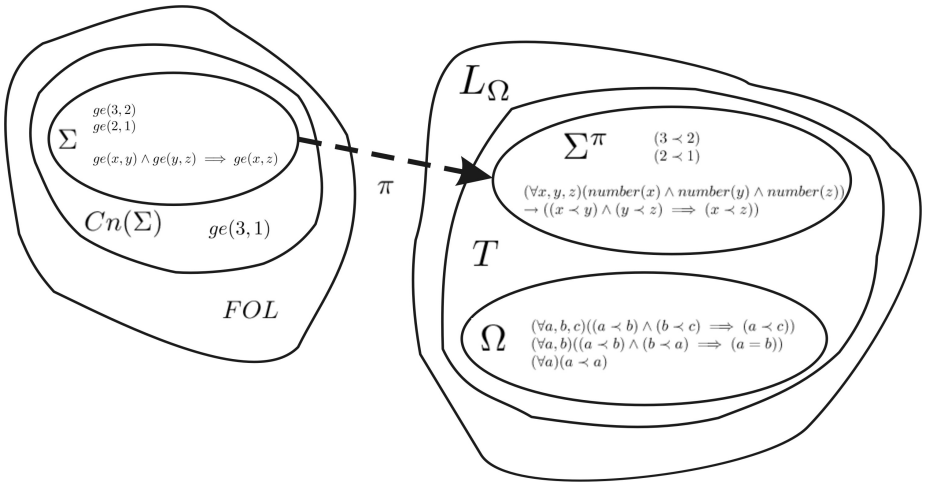


Figure 7.2: Interpretation of Σ with respect to L_Ω

these requirements. The sentences to be translated are class names in the different classifications, the logical rendering is achieved through the use of a terminological language and the interpretation is given by the definition $d(C_i)$ of a class names C_i. The shared ontology corresponds to Ω. These analogies imply that we can use logical deduction as defined in 7.1 in order to perform translations.

In the following, we first define the translation task with respect to information systems and classifications. We then show how reasoning in a terminological language can be used in order to perform partial translations from one system into the other. Based on these transformations, we introduce an approach for information filtering that is based on re-writing Boolean queries across heterogeneous classifications.

7.2.1 Ontology Heterogeneity

Ontologies can differ in many ways [Visser et al., 1997]. We will not try to discuss them in general. We will rather give an example of ontologies that even though they describe the same domain of interest represent very different conceptualizations of that domain. We start with a simple ontology that discriminates animals into domestic, foreign and production animals and contains some kinds of animals that fall under one or more of these categories (compare figure 7.3).

Now consider an ontology that describes classes of animals in the way a child would possibly categorize them (compare figure 7.4). The main distinctions made in this ontology are pets, farm animals and zoo animals. These distinction

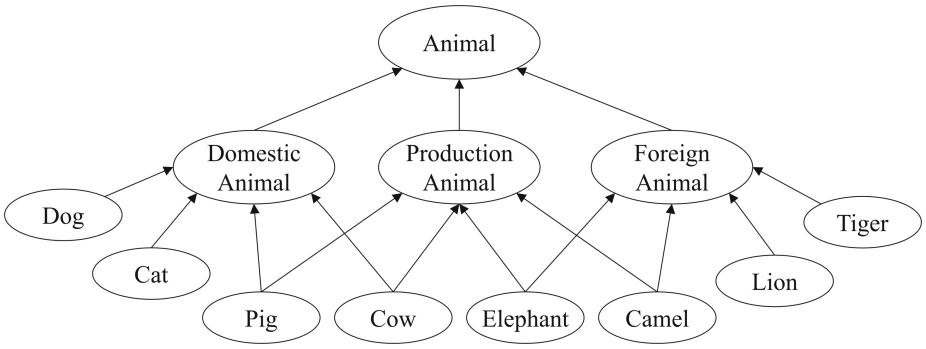


Figure 7.3: An ontology of animals

are based on the experience of a child that some animals are kept at home, at farms or in zoos.

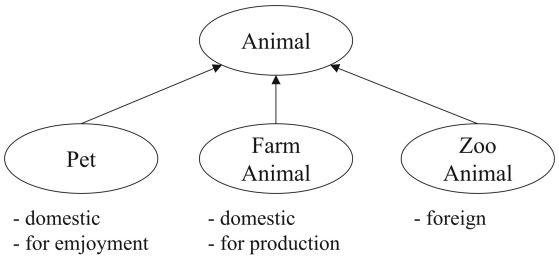


Figure 7.4: A 'childish' ontology of animals

While both ontologies do not share any class except for the general class animal, it should be possible to establish a relation between the two. Using common world knowledge and the informal descriptions of the classes in figure 7.4 we can conclude that 'Pet' should be a subclass of 'Domestic Animal' and include 'Cat' and 'Dog'. 'Farm Animal' should be a subclass of 'Domestic Animal' and include 'Cow' and 'Pig'. Finally, 'Zoo Animal' should be subsumed by 'Foreign Animal' and contain all the subclasses of 'Foreign Animal' shown in figure 7.4

7.2.2 Multiple Systems and Translatability

In order to get a clearer notion of the problem to be solved, we give an abstract definition of an information source in terms of a set of information items that are classified according to a source ontology. This general notion of an information

source covers Web-based information system like the one discussed in the last chapter. An information item corresponds to a single Web page that has been classified according to a source ontology. In the case of a conventional database, we can think of single rows in a database table as an information item. The connection to a source ontology can be given by a corresponding reference in the Data Dictionary that may also contain the ontology itself.

Definition 7.1 (Information Source) *An Information source is a tuple* $\langle O, I, M\rangle$ *where* $O = \langle S, C, L, d\rangle$ *is a source ontology with shared ontology* S*, a set of class names* C *and a mapping* d *that assigns a class definition in the terminological language* L *over terms from* S *to every class name from* C*,* I *is a set of information items and* $M : I \times C$ *is a membership relation relates information items to classes of the source ontology.*

Building on this abstract view on an information source, we can define the problem of integrating the classifications employed in two different systems. Roughly speaking the task is to extend the membership relation M_1 of an information source IS_1 by an additional relation M' that relates the information items of a second information source IS_2 according to the source ontology of IS_1. In order to be able to reason about instances of classes as well, we extend the semantics of a terminological language in a straightforward way, by assuming that the assignment mapping $\mathcal{A}$ does not only apply to class names but also to instances. We define that:

$$x^{\mathcal{A}} \in W \text{ for every } x \in I \tag{7.2}$$

Based on this extended notion of assignment, we can also define the interpretation of individuals:

$$x^{\Im,\mathcal{A}} =_{def} x^{\mathcal{A}} \tag{7.3}$$

The notion of an interpretation of individuals allows us to reason about the membership of instances to classes, denoted as $x : C$. We define membership as follows:

$$x : C \iff x^{\Im,\mathcal{A}} \in C^{\Im,\mathcal{A}} \tag{7.4}$$

Using this definition of the semantics of individuals with respect to an ontology, we can define the translation problem. we have to solve as follows:

Definition 7.2 (Integration Problem) *Let* $IS_1 = \langle\langle S_1, L_1C_1, d_1\rangle, I_1, M_1\rangle$ *and* $IS_2 = \langle\langle S_2, L_2C_2, d_2\rangle, I_2, M_2\rangle$ *be information sources, then a bilateral integration problem is equivalent to finding a membership relation* $M : I_1 \cup I_2 \times C_1$ *such that for all* $x \in I_2 \cup I_2$ *and* $c_i \in C_1$*:*

$$(x, c_i) \in M \textit{ iff } x : d_1(c_i)$$

In order to generate this new relation M' we have to rely on the semantics of both information sources that is given by their source ontology. In general, we can not assume that both information sources use the same source ontology. We cannot even assume that the source ontologies of both information sources are comparable at all. If we want to make assertions about the relation of the ontologies of two information sources, we have to ensure that we can perform terminological reasoning across these ontologies. This in turn is given, if both ontologies share the interpretation $\Im$ of concept terms and the assignment mapping from primitive terms into an abstract domain $\mathcal{A}$ (compare defintions 4.5 and 4.6). We can ensure this if both source ontologies are based on the same shared ontology and use terminological languages that are in the same model-based family of languages.

Definition 7.3 (Comparability) *Let* $IS_1 = \langle\langle S_1, C_1, L_1, d_1\rangle, I_1, M_1\rangle$ *and* $IS_2 = \langle\langle S_2, C_2, L_2, d_2\rangle, I_2, M_2\rangle$ *be information sources, then these information sources are said to be comparable, iff*

1. $S_1 = S_2$
2. *there is a model-based family of languages* $\mathcal{L}_\mathcal{F}$ *according to definition 4.12 such that* $L_1 \in \mathcal{L}_\mathcal{F}$ *and* $L_2 \in \mathcal{L}_\mathcal{F}$

Proposition 6 (Unique Interpretation) *If two information sources are comparable, their source ontologies shared a unique interpretation.*

Proof 6 *In order to proof that two ontologies share the same models we have to show that the interpretation is defined uniquely for the two models. This includes the the mappings* $\Im$ *and* $\mathcal{A}$*.* $\mathcal{A}$ *is defined with respect to a specific set of atomic terms. From the definition, we get that these terms are shared in both systems. Therefore* $\mathcal{A}$ *is uniquely defined. With respect to* $\Im$ *the same result directly follows from proposition 4 on page 61).*

This result in fact gives us the possibility to reason across the source ontologies of both information sources. The reason for this is straightforward: The use of the same shared ontology provides us with a unique interpretation of the primitive terms defined therein. These terms are used in the source ontologies to define different sets of more complex concepts, even potentially using different terminological languages. However, we can define transformations of both languages involved into a joint language while preserving interpretation.

7.2.3 Approximate Re-classification

The comparability criterion given above allows us to reason across source ontologies, however, the definitions included in the different ontologies will often be similar but not equivalent. This might lead to a situation, where we are not able to decide whether an instance really belongs to a certain class in a different system or not. However, we can identify cases where we are able to decide whether an instance from a remote information source definitely belongs

to certain class or definitely not belongs to a certain class.

Consider the situation where we want to classify an information item from an information source IS_2 into the local ontology of IS_1 by computing M. The only information we have about x is its classification M_2 with respect to the source ontology of IS_2. In order to make use of this information, we have to determine the relation between possible classifications of x in IS_1 and the source ontology of IS_2. In this context, we can use subsumption testing in order to determine hypotheses for M with respect to IS_2 by computing the class hierarchy for $C_1 \cup C_2$ using the definitions of individual classes (provided that the encoding languages belong to the same family of languages).

As the classes in the hierarchy form a partial order, we will always have a set of direct super- and a set of direct subclasses of c_1. We can use these classes as upper and lower approximation for c_1 in IS_2:

Definition 7.4 (Upper Approximation) *Let $IS_1 = \langle\langle S_1, L_1 C_1, d_1\rangle, I_1, M_1\rangle$ and $IS_2 = \langle\langle S_2, L_2, C_2, d_2\rangle, I_2, M_2\rangle$ be information sources and $c \in C_1$ a class from IS_1, then a class $c_{lub} \in C_2$ is called a least upper bound of c in IS_2, if the following assertions hold:*

1. $d_1(c) \sqsubseteq d_2(c_{lub})$
2. $(\exists c' \in C_2 \text{ such that } d_1(c) \sqsubseteq d_2(c')) \implies (d_2(c_{lub}) \sqsubseteq d_2(c'))$

The upper approximation $lub_{IS_2}(c)$ is the set of all least upper bounds of c in IS_2.

Definition 7.5 (Lower Approximation) *Let $IS_1 = \langle\langle S_1, L_1, C_1, d_1\rangle, I_1, M_1\rangle$ and $IS_2 = \langle\langle S_2, L_2, C_2, d_2\rangle, I_2, M_2\rangle$ be information sources and $c \in C_1$ a class from IS_1, then a class $c_{glb} \in C_2$ is called a greatest lower bound of c in IS_2, if the following assertions hold:*

1. $d_2(c_{glb}) \sqsubseteq d_1(c)$
2. $(\exists c' \in C_2 \text{ such that } d_2(c') \sqsubseteq d_1(c)) \implies (d_2(c') \sqsubseteq d_2(c_{glb}))$

The lower approximation $glb_{IS_2}(c)$ denotes the set of all greatest lower bounds of c in IS_2.

The rational of using these approximations is that we can decide whether x is a member of the classes involved based on the relation M_2. This decision in turn provides us with an approximate result on deciding whether x is a member of c_1, based on the following observation:

- If x is member of a lower bound of c_1 then it is also in c_1
- If x is not member of all upper bounds of c_1 then it is not in c_1

In [Selman and Kautz, 1996] Selman and Kautz propose to use this observation about upper and lower boundaries for theory approximation. We adapt the proposal for defining an approximate classifier $M' : I_2 \times C_1 \rightarrow \{0, 1, ?\}$ in the following way:

Definition 7.6 (Approximate Re-Classification) *Let* $IS_1 = \langle\langle S, C_1, L_1, d_1\rangle, I_1, M_1\rangle$ *and* $IS_2 = \langle\langle S, C_2, L_2, d_2\rangle, I_2, M_2\rangle$ *be information sources and* $x \in I_2$ *then for every* $c_1 \in C_1$ *we define* M' *such that:*

- $M'(x, c_1) = 1$ *if* $x : \left(\bigvee_{c \in glb_{IS_2}(c_1)} d_2(c) \right)$
- $M'(x, c_1) = 0$ *if* $x : \neg \left(\bigwedge_{c \in lub_{IS_2}(c_1)} d_2(c) \right)$
- $M'(x, c_1) = ?$, *otherwise*

Where the semantics of disjuction and conjunction is defined in the obvious way using set union and intersection (compare definition 4.6.

Based on the observation about the upper and lower bounds, we can make the following assertion about the correctness of the proposed approximate classification:

Proposition 7 (Correctness of the Approximation) *The approximation from definition 7.6 is correct in the sense that:*

1. *If* $M'(x, c_1) = 1$ *then* $x^{\Im,\mathcal{A}} \in d_1(c_1)^{\Im,\mathcal{A}}$
2. *If* $M'(x, c_1) = 0$ *then* $x^{\Im,\mathcal{A}} \notin d_1(c_1)^{\Im,\mathcal{A}}$

Using the definition of upper and lower bounds the correctness of the classification can be proven in a straightforward way:

Proof 7 *(1) If the classification returns* $M'(x, c_1) = 1$ *then* $x : (\bigvee_{c \in glb_{IS_2}(c_1)} d_2(c))$. *Using definition 7.5 we get that for all* c *we have* $d_2(c) \sqsubseteq d_1(c_1)$ *and therefore also* $(\bigvee_{c \in glb_{IS_2}(c_1)} d_2(c)) \sqsubseteq d_1(c_1)$ *(by set theory). Using the definition of subsumption we can conclude that* $x^{\Im,\mathcal{A}} \in d_1(c_1)^{\Im,\mathcal{A}}$.

(2) Using definition 7.4 we deduce that for all c *we have* $d_1(c_1) \sqsubseteq d_2(c)$ *and therefore* $d_1(c_1) \sqsubseteq \bigwedge_{c \in lub_{IS_2}(c_1)} d_2(c)$. *This means that* $x^{\Im,\mathcal{A}} \in d_1(c_1)^{\Im,\mathcal{A}}$ *only if* $x^{\Im,\mathcal{A}} \in (\bigwedge_{c \in lub_{IS_2}(c_1)} d_2(c))^{\Im,\mathcal{A}}$. *However if the classification returns* $M'(x, c_1) = 0$ *then* $x : \neg(\bigwedge_{c \in lub_{IS_2}(c_1)} d_2(c))$ *which is equivalent to* $x^{\Im,\mathcal{A}} \notin (\bigwedge_{c \in lub_{IS_2}(c_1)} d_2(c))^{\Im,\mathcal{A}}$. *Therefore we also have* $x^{\Im,\mathcal{A}} \notin d_1(c_1)^{\Im,\mathcal{A}}$.

This results provides us with the possibility to include many of the information items from remote systems into an information source in such a way, that we get a semantic description of the item we can use for information management. Another interesting application of this approach, namely information filtering is described in the next section.

7.3 Information Filtering

The translation approach described in the last section allows us to include arbitrary information items into our own system, provided that we are able to re-classify it using the approximate method we introduced. However, in most cases we are not interested in the whole information of a remote system, but only in information about a specific topic. The approach of first trying to translate the whole information source in most cases leads to a significant overhead, especially when we consider the amount of information available on the World Wide Web. We therefore strive for methods that allow us to pre-select relevant information from remote systems by posing specific queries to these systems.

As the major structuring method we use in this work is the classification of information entities according to the source ontology, we want to use the semantics defined in the ontology also as a basis for selecting information from remote systems. For this purpose we propose to use Boolean queries over concept names from the classification hierarchy. However, if we want to use the vocabulary provided by the ontology of our information source, we again face the problem of heterogeneity with respect to the ontologies used in other systems. We show how we can use approximate re-classification in order to translate the queries we want to post to remote systems in such a way that we can guarantee that all returned information items indeed satisfy the query expression.

Due to the approximate nature of the re-classification, we will not be able to guarantee that all interesting information items are actually returned on a query, because we just do not have an appropriate vocabulary for stating queries in such a way that they cover all information items from the remote system. So our approach while not being able to provide maximal recall, it guarantees maximal precision with respect to the semantics of the query.

7.3.1 The Idea of Query-Rewriting

Assume that we want to post a query formulated using the ontology from figure 7.4 to an information source that has been classified according to the ontology in figure 7.3. In order to answer this query, we have to resolve the heterogeneity discussed above. The use of a shared ontology in combination

with a definition of the classes in both ontologies enable us to do this. As an example we take the following query (Animal $\wedge \neg$(Farm Animal)). This query cannot be directly answered, because the term 'Farm Animal' is not understood, but we know what are the characteristic properties of 'Zoo Animal' and can compare them with the definitions of classes in the other ontology.

As described in the introduction, the idea of our approach is to re-write this query in such a way that it covers the same set of answers using terms from the other ontology. In general, an exact re-writing is not possible because the concepts of our ontology do not have corresponding concepts. In this case, we have to look for re-writings that approximate the query as closely as possible. Re-writings that are an upper approximation of the original query are know from the database area as *minimal subsuming mappings* [Chang and Garcia-Molina, 2001]. While in the area of databases upper approximations are often used in combination with an additional filter that removes irrelevant results, our approach aims for correctness rather than for completeness and therefore uses a lower approximation.

The idea of the re-writing is the following. Based on the formal definitions of the classes in both ontologies, we can find those concepts in the ontology of figure 7.3 that are most closely related to a query concept. Taking a concepts from our query, we can for example decide that 'Domestic Animal' and 'Production Animal' are upper approximations for 'Farm Animal' while 'Cow' and 'Pig' are lower approximations. Using these concepts, we can define lower boundaries for 'Farm Animal' (Cow $\vee$ Pig) and use this expression instead of the original concept still getting correct results. In our example, however, the concept occurred in a negated form. In order to return a correct result, we therefore cannot use the lower bound because not all irrelevant resources might be excluded. Based on the considerations made above we can replace the concept 'Farm Animal' within the scope of the negation by its upper bound (Domestic Animal $\wedge$ Production Animal). Using this rewriting, we get the following query that can be shown to return only correct results: (Animal $\wedge \neg$(Domestic Animal $\wedge$ Production Animal).

The situation becomes slightly more complicated if complex expressions occur in the scope of a negation. An example is the following query: $\neg$(Pet $\vee$ Farm Animal). In this case we first have to convert the query into negation normal form where negation only applies to atomic concepts. In negation normal form the above query will be of the form ($\neg$Pet $\wedge \neg$Farm Animal). Using upper and lower bounds this query translates to $\neg$Domestic Animal $\wedge \neg$(Domestic-Animal $\wedge$ Production Animal). This query normalizes to ($\neg$Domestic Animal$\wedge \neg$Production Animal) which in our example only includes the classes 'Lion' and 'Tiger'.

7.3.2 Boolean Queries

In order to apply the idea of approximate re-classification to information filtering, we first have to define the type of filtering expressions we want to use. As mentioned in the introduction, we use the Boolean query model widely used in information retrieval and filtering. In information retrieval, Boolean queries consist of keywords that are combined by Boolean operators. The assignment of information items to concepts of an ontology as proposed in the last chapter enables us to use concept names instead of keywords. The resulting notion of a Boolean concept query is defined as follows:

Definition 7.7 (Boolean Concept Query) *Let $IS = \langle\langle S, C, L, D\rangle, I, M\rangle$ be an information source, then a Boolean query is formed by a legal query expression that are defined in the following way:*

1. *every $c \in C$ is a legal query expression*
2. *if e is a legal query expression then $\neg e$ is also a legal query expression*
3. *if e_1 and e_2 are legal query expressions, then $e_1 \wedge e_2$ and $e_1 \vee e_2$ are also legal query expressions.*

The advantage of using concept names instead of keyword is the possibility of defining a clear semantics of a query that makes it possible to reason about the query result in the framework of terminological languages. By defining the semantics of a Boolean concept query on the basis of the semantics of the concept contained therein, we get a direct connection between queries and the underlying ontology. This is of particular interest for the case where queries are not posed by human users, but by computer programs, because the semantics of the queries enables the system to precisely interpret the returned result.

Definition 7.8 (Query semantics) *Let $IS = \langle O, I, M\rangle$ be an information source. The semantics $Q^{\mathcal{I}}$ of a query Q is defined by an interpretation mapping $\mathcal{I}$ into the abstract description model of O in the following way:*

1. $c^{\mathcal{I}} = d(c)^{\Im,\mathcal{A}}$
2. $(\neg e)^{\mathcal{I}} = W - e^{\mathcal{I}}$
3. $(e_1 \wedge e_2)^{\mathcal{I}} = e_1^{\mathcal{I}} \cap e_2^{\mathcal{I}}$
4. $(e_1 \vee e_2)^{\mathcal{I}} = e_1^{\mathcal{I}} \cup e_2^{\mathcal{I}}$

The reason for relating queries to ontologies on a semantic level is the possibility to use terminological reasoning for determining the query result. We can treat the query as a concept expression in the ontology and classify it with respect to the other concepts therein. In especially, we can determine those concepts in the ontology that are subsumed by the query:

Definition 7.9 (Subsumed Concepts) *A concept C is said to be subsumed by a query q if $d(C)^{\Im,\mathcal{A}} \subseteq Q^{\mathcal{I}}$ we denote this fact by $C \sqsubseteq Q$.*

On the other hand, what we are interested in are those information items that are members of the concept expression that is equivalent to the query. These items we refer to as the *query result* are formally defined as follows:

Definition 7.10 (Query Result) *Let $IS = \langle\langle S, C, L, D\rangle, I, M\rangle$ be an information source and Q be a Boolean query over IS then the result of Q is given by:*

$$\{x \in I | x^{\Im,\mathcal{A}} \in Q^{\mathcal{I}}\}$$

We denote the fact that an instance x belongs to the result of a query Q by $x : Q$

Subsumed concepts directly provide us with the result to a terminological query, because the union of their members is exactly the query result we are looking for. As information items are explicitly assigned to concepts in the ontology, the task of computing the query results reduces to looking up the members of the subsumed classes.

Proposition 8 *An information item x is in the result of a query Q if*

$$M_2(x, C) \wedge d(C) \sqsubseteq Q$$

This proposition directly follows from the definition of subsumption and membership. Though being trivial, we include it for the sake of completeness

Proof 8 *By definition we have $x^{\Im,\mathcal{A}} \in d(C)^{\Im,\mathcal{A}}$. From proposition 8 we get $x^{\Im,\mathcal{A}} \in Q^{\mathcal{I}}$, because $d(C) \sqsubseteq Q$ and therefore $d(M(x))^{\Im,\mathcal{A}} \subseteq Q^{\mathcal{I}}$.*

These considerations justify the use of description logic reasoners for answering Boolean concept queries.

7.3.3 Query Re-Writing

In the last section we described how information filtering with Boolean queries can be implemented using terminological reasoning. We showed that in our framework the problem of filtering relevant information can be reduced to subsumption reasoning. However, the approach assumed that the concept names used in the Boolean query are taken from the ontology of the information source that is queried, because the definitions of those concepts have to be known in order to determine subsumption relations. At this point, the re-classification results given in the last section come into play. The idea is to approximate the meaning of concepts in a query by its re-classification, i.e. by the upper and lower bounds in the other system (compare section 7.2.3).

The adaption of a query to remote systems can be done in a three step process:

1. **Normalization:** the original query is transformed into negation normal form (see definition 7.11).

2. **Re-writing:** the concept names in the query are replaced by their approximations in the remote source. (this is done for each remote source individually).

3. **Classification:** the re-written query is classified into the ontology of the remote source and instances of subsumed concepts are returned as result.

The transformation to negation normal form is necessary in order to decide whether a concept name has to be replaced by its lower or its upper bound. As argued in the first section of this chapter, negated concepts have to be replaced by their upper bound and non-negated ones by their lower bound in order to ensure the correctness of the query result. The negation normal form supports this process by revealing which concept names are negated and which not.

Definition 7.11 (Negation Normal Form) *A query is said to be in negation normal form if negations only apply to concept names $c \in C$ and not to compound expressions.*

Every Boolean query can easily be transformed into negation normal form using the following equalities:

$$\neg(e_1 \wedge e_2) \equiv \neg e_1 \vee \neg e_2 \tag{7.5}$$

$$\neg(e_1 \vee e_2) \equiv \neg e_1 \wedge \neg e_2 \tag{7.6}$$

Once we have transformed the query, re-writing can be done locally on the concept names using the least upper and greatest lower bounds that have already been discussed in the last section:

Definition 7.12 (Query Re-Writing) *The re-writing of a query Q in negation normal form over concepts c_i from an information source IS_1 to a query Q' over concepts from another information source IS_2 as follows:*

- *replace every non negated concept name c by:* $\bigwedge_{c' \in lub_{IS_2}(c)} c'$
- *replace every negated concept name c by:* $\bigvee_{c' \in glb_{IS_2}(c)} c'$

The rewriting and execution of a query can easily be implemented using the Description Logic System RACER [Haarslev and Moller, 2001]. We can compute the re-writing using Algorithm 1. The input for the algorithm is the query to be re-written, the class names in C_2 and a terminological knowledge base including the definitions of the concepts in C_1 and C_2 as well as the shared ontology.

As the re-writing builds upon the approximations discussed in the last section we can guarantee that the result of the query is correct. Moreover, we can use

Algorithm 1 Rewrite-Query

Require: A Boolean query in negation normal Form: Q
Require: A list of class names: N
Require: A terminological knowledge base T
  racer.in-tbox(T)
  for all t is an atomic term in Q **do**
    if t is negated **then**
      $B[t]$:= racer.directSupers(t)
      $B'[t] := B[t] \cap N$
      $Q(t) := (c_1 \wedge \cdots \wedge c_n)$ for $c_i \in B'[t]$
    else
      $B[t]$:= racer.directSubs(t)
      $B'[t] := B[t] \cap N$
      $Q(t) := (c_1 \vee \cdots \vee c_n)$ for $c_i \in B'[t]$
    end if
    r(Q) := **proc** Replace t in Q by $Q(t)$
  end for
  return r(Q)

subsumption reasoning in order to determine this result. To be more specifically, a resource x is indeed a member of the query concept if membership can be proved for the re-written query.

Proposition 9 (Correctness of re-writing) *The notion of query re-writing defined above is correct in the sense that:*

$$x : Q' \implies x^{\Im,\mathcal{A}} \in Q^{\mathcal{I}}$$

Proof 9 *From proposition 7 we get that* $x : (\bigwedge_{c' \in lub_{IS_2}(c)} c')$ *implies* $x^{\Im,\mathcal{A}} \in c^{\Im,\mathcal{A}}$ *and that* $x : \neg(\bigvee_{c' \in glb_{IS_2}(c)} c')$ *implies* $x^{\Im,\mathcal{A}} \notin c^{\Im,\mathcal{A}}$*. This establishes the correctness of re-writing for atomic queries, i.e. non negated and negated concept names. Assuming a queries in negation normal form, it remains to be shown that the correctness is preserved for conjunctions and disjunctions of negated and non-negated concept names.*

We proof the overall correctness by induction over the definition of legal query expressions. By induction hypothesis (established above) we have $x \in e_1' I \implies x\Im, \mathcal{A} \in e_1^{\mathcal{I}}$ *and* $x : e_2' \implies x^{\Im,\mathcal{A}} \in e_2^{\mathcal{I}}$*. For the induction step we have to distinguish the following cases:*

(case 1: $q = e_1 \wedge e_2$*) as* $x\Im, \mathcal{A}$ *is in* $e_1^{\mathcal{I}}$ *and* $e_2^{\mathcal{I}}$ *by induction hypothesis it is also in* $e_1^{\mathcal{I}} \cap e_2^{\mathcal{I}}$ *and therefore in* $q^{\mathcal{I}}$*.*

(case 2: $q = e_1 \vee e_2$) as $x\Im, \mathcal{A}$ is in $e_1^{\mathcal{I}}$ or in $e_2^{\mathcal{I}}$ by induction hypothesis it is also in $e_1^{\mathcal{I}} \cup e_2^{\mathcal{I}}$ and therefore in $q^{\mathcal{I}}$.

The results proven in this section provide us with a tool to filter information items according to Boolean expressions across heterogeneous information sources provided that they use the architecture described in the second part of this thesis. We consider this a great advantages because the search for interesting information no longer has to be based on plain keywords whose meaning is not precisely defined leading to problems concerning precision and recall.

Unfortunately, proving the correctness of the approximation says nothing about the quality of the approximation. In the worst case, the upper and lower boundaries of concepts in the other hierarchy are always $\top$ and $\bot$ respectively. In this case the translated query always returns the empty set as result. We were not able to investigate the quality of approximations on theoretical level, however, we can provide some rules of thumb that can be used to predict the quality of an approximation:

Depths of hierarchies: The first rule of thumb, we can state is that deeper class hierarchies lead to better approximations. For hierarchies of depth one it is easy to see that we will not be able to find good upper and lower bounds. We can also assume that deeper hierarchies provide finer grained distinctions between concepts that in turn often produce closer approximations.

Degree of overlap: Our approach assumes a shared vocabulary for building class definitions, however, we cannot guarantee that different systems indeed use the same parts of this shared vocabulary. Therefore, the actual overlap of terms used in the existing definitions that are compared is important for predicting the quality of approximations. In general, we can assume that a high degree of overlap leads to better approximations.

Both criteria used in the rules of thumb above strongly depend on the application and on the creator of the corresponding models. At least for the degree of overlap, we can assume that hierarchies that are concerned with the same domain of interest will share a significant part if the vocabulary, thus enabling us to compute reasonable approximations.

Conclusions

While the idea of maximizing precision and recall by using concept expressions in Boolean queries is an appealing idea, the practical application in information sharing suffers from the fact that there will not be 'the one' ontology that is used to classify information. We will rather face a situation, where a multitude of classification hierarchies organize different or even the same

information according to different discrimination principles. A successful information filtering approach will have to make use of as many of these ontologies as possible. This claim raises the problem of comparing different ontologies.

Based on the logical model described in chapters 2 and 3 and a shared vocabulary we can perform terminological reasoning across different ontologies. A problem that still persists is the fact that different source ontologies will almost never contain concepts with exactly the same meaning. Therefore we cannot exactly map concepts from two information source on each other. However, following the idea of theory approximation we can use upper and lower bounds to get mapping results that can be proven to be correct in the sense that they they provide maximal precision, but not completeness (i.e. the recall cannot be guaranteed to be maximal).

This approximate mapping approach can also be used to contribute to the problem of finding relevant information in different information sources. We can answer Boolean queries over concept names by replacing unknown concept names by their lower bound in the corresponding source ontology. The resulting query can be processed in the context of the remote information source delivering a query result that can be proven to be a correct approximation of the intended result.

We conclude that approximation techniques for processing queries and for logical reasoning in general are important in weakly structured and heterogeneous environments such as the World Wide Web because they can be used to partly overcome semantic heterogeneity that is omnipresent.

Chapter 8

The BUSTER System

The goal of this chapter is to give evidence for the practical applicability of the models and methods presented in this thesis. After having proposed a logical framework and an architecture for representing information semantics as well as the possibility to generate metadata based on this framework and methods to reason about information contents, we now present our prototype implementation of many of the ideas of this thesis. We present the implementation focusing on those aspects that directly relate to our work, namely methods for information filtering and semantic mapping.

Acknowledgement: The description of the system found in this chapter partly occurs in [Neumann et al., 2001] with is co-authored by Holger Neumann, Gerhard Schuster, Ubbo Visser, Thomas Voegele and Holger Wache. The description of the implementation is taken from [Stuckenschmidt, 2001]

The methods developed in this thesis are meant to be not just of theoretical interest, but to provide at least partial solutions for real world problems. For this reason, we implemented the 'Bremen University Semantic Translator' (BUSTER), a system meant to provide an intelligent middleware for information sharing. We envision that the BUSTER system is used by many different applications like search engines, e-commerce platforms or corporate memories in order to access heterogeneous and distributed information resources. For this purpose, the BUSTER system provides two subsystems, one for information filtering and one for information integration. These subsystems are mainly independent of each other and can be accessed by clients over the World Wide Web (see figure 8.1).

In this chapter, we describe the basic ideas of the BUSTER system and give an illustration of its functionality. We will especially focus on those parts of the system that implement methods and infrastructure presented in this thesis. We implemented some of the results presented in this thesis in the

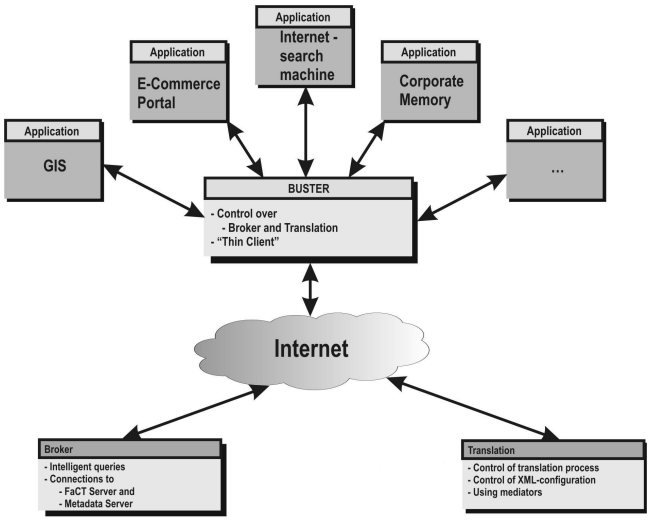


Figure 8.1: BUSTER - Intelligent Middleware for Information Sharing

BUSTER system in order to solve the two main tasks: information filtering and semantic translation of information content. The inference module provides terminological reasoning facilities for the information filtering subsystem and re-classification services for the information integration subsystem (see figure 8.2)

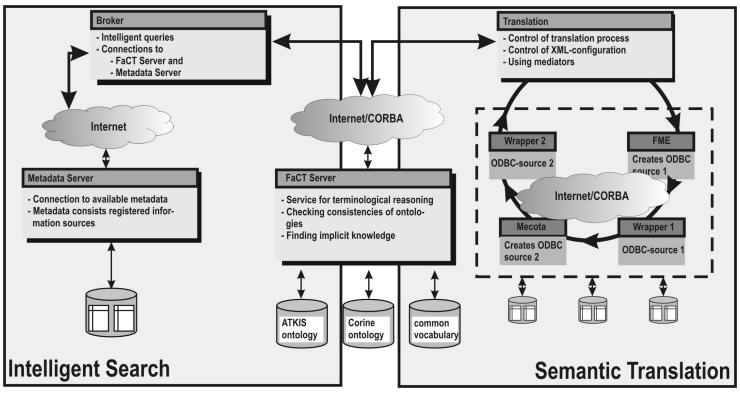


Figure 8.2: Information Filtering and Integration in BUSTER [Neumann et al., 2001]

In the following, we briefly describe the reasoning module. Afterwards we illustrate how the infrastructure described in this thesis and the reasoning module is used to filter and integrate information.

8.1 The BUSTER Reasoning Module

This section gives a brief introduction to the implementation of the BUSTER inference module [Stuckenschmidt, 2001]. The module implements some of the results of this thesis by imposing higher-level inference methods on top of the description logic reasoner RACER [Haarslev and Moller, 2001]. The connection to the reasoner is established by calling methods of the generic reasoning client that is included in the RACER system. The module has two main functions with respect to semantic translation:

1. Providing the infrastructure for managing multiple logical theories and providing programming interfaces to this infrastructure.

2. Implementing the top-level control flow of the semantic translation process based on the functionality provided by description logic reasoners.

For these purposes the module contains a set of classes the most important of which will be described in the following.

8.1.1 Theory Management

The central concept for organizing and managing knowledge bases is the class `theory`. It refers to the general notion of a logical theory and contains a definition of the vocabulary used (`signature`), a set of definitions in terms of logical sentences and theorems which are facts that logically follow from the definitions. In order to facilitate reasoning in a given context determined by a theory, each instance of the theory class is also equipped with an instance of the RACER client and therefore owns its own inference engine. In the following, we describe the classes implementing the elements of a theory in more detail.

Signature: The class `signature` implements a structure that maintains the names of concepts, relations and individuals mentioned in the theory it is attached to. The corresponding names are automatically inserted into the structure when a new concept, relation or instance is defined. The signature can be used to check, whether two source ontologies use the same shared ontology (see definition 5.2 and can therefore be used to check whether two ontologies are comparable (see definition 7.3).

Theorem: Theorems define concepts, instances and individuals. A theorem consists of an object which is essentially the name to be inserted into the signature and a sentence that further defines the object. Theorems can be distinguished in `definitions` and `mappings`. For the convenience of defining terminological theories the system contains the following subclasses of the class theorem:

- *PrimClassDef:* the definition of a primitive concept

- *ClassDef:* the definition of a defined concept
- *RoleDef:* the definition of a primitive Role
- *InstanceDef:* the definition of a individual
- *Equivalence:* a mapping stating the equivalence of a concept name and an expression in another theory
- *Implication:* a mapping stating the that an expressions in another theory follows from a class name

Theorems correspond to class definitions in terms of axioms in a terminological languages (compare section 4.2). For the sake of simplicity, the BUSTER system uses DAML+OIL (section 3.2 as the only language to encode class definitions.

Taxonomy Node: Taxonomy nodes are used to store the lattice structure of a classified theory in a way that makes efficient access possible. A taxonomy node stores the name of the corresponding concept, a list of its super- and subclasses as well as a set of individuals belonging to the concept. Each theory contains a hash-table with taxonomy nodes for all concepts in its signature indexed by the concept name. Perceiving classification as the main inference task in description logic theories, these nodes correspond to theorems that can be stated about the theory. The hash-table is filled by calling the `compile`-method of a theory. Afterwards simple reasoning tasks can be performed without using the reasoner. This is especially important when dealing with very large theories. In the system, this data structure contains the class hierarchy of an information source after it has been verified and completed in the way described in section 6.4.1.

8.1.2 Interpretations and Translations

Theories are connected by interpretations. An interpretation is a theory that defines mappings between two other theories denoted as `source` and `target` respectively and are stored as attributes of an instance object. Interpretations are used to describe the assignment d of class expressions to the classes used in an information source (compare definition 5.3). Interpretations inherit all methods from theories, especially the methods for inserting definitions are used in order to define mappings. It is assumed that only mappings are defined in Interpretations, however, this is not enforced by the system. Inserting other kinds of theorems is possible, but may lead to unexpected results in the translation process. The interpretations class specifies the method `apply` that when called applied the interpretation on the source theory and merges the result with the target theory. The resulting theory is returned as a result.

While the application of an interpretation is a very generic way of relating theories, the special process of translating concept names from one vocabulary

into another is implemented in the class `Translator`. This class is also designed as a container for storing theories and interpretations, however, yet without methods for managing these components. The `translator` class contains lists of theories and interpretations and implements two methods for translating between theories that are both based on the notion of re-classification (definition 7.6). The system uses a simplified variant of the query rewriting algorithm (see Algorithm 1 on page 142) in order to replace concept names from one into concept names from another source ontology.

At the implementation level, the translation methods are based on interpretations rather than theories, because in order to be translatable into another vocabulary both theories have to be interpreted in the same shared theory. Therefore, the method `translate` gets the name of a concept and two interpretations as input. It checks

1. whether the concept name is in the source theory of the first interpretation
2. whether the target theories of the two interpretations are identical

If both conditions hold, both interpretations are applied and the result merged forming a unified theory. This theory is compiled an the result stored in the corresponding structure which is then used to determine those super-classes of the concept name that are from the target theory. A List of these concepts (i.e. the names) are returned. In order to translate a complete theory, the translator class also contains the method `translateAll` that returns a hash-table of the translations of all concept names in the source theory. The table is indexed by the names of the concepts that are translated. The translations are arrays containing possible translations, where the most specific translation is the last element in the array.

8.2 Information Filtering

The first step in information sharing with the BUSTER system is the selection of an information source to be incorporated in the existing information of the user. The system supports this selection step by content-based filtering methods that implement concept-based queries as described in section 6.4.2 and 7.3.2. The user is asked to select the type of information she is interested in by choosing the name of a concept from the shared ontology (see section 5.2.2). Based on the relations this concept may have to other concepts (these are also specified in the shared ontology) a user interface is created dynamically that helps the user to further refine the specification of the corresponding concept. Figure 8.3 shows the interface with an example query.

In principle the definition of the concept that specifies the information needs of a user could be further refined by restricting the slot-fillers of its relations using arbitrary concept expressions. In BUSTER, we decided to restrict the

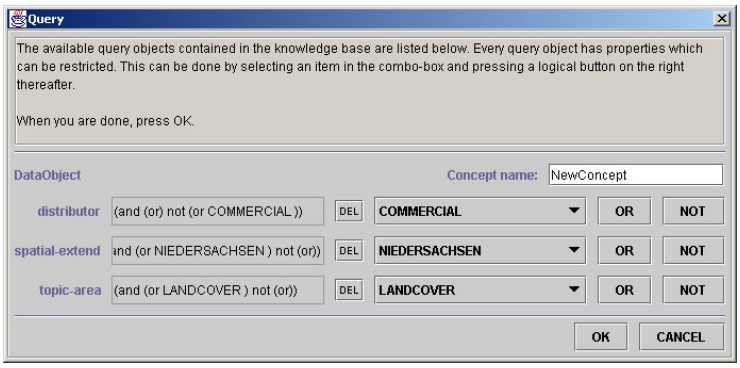


Figure 8.3: User interface for composing queries from a pre-defined vocabulary

language that can be used for the refinement in different ways in order to help users not familiar with description logics to come up with useful definitions that can be matched against the descriptions of available information sources.

Type of restriction: We only allow existential restrictions, because the implication that underlies universal restrictions is often mis-interpreted by users. Further, using only existential restrictions in the description of the information sources as well, we get better classification results, because the interaction between existential and universal restrictions is more complicated and leaves space for cases where subsumption is likely, but cannot be proven.

Logical operators: We only allow the types of fillers to be described by an expression of the form:

$$(C_1 \vee \cdots \vee C_n) \wedge \neg(D_1 \vee \cdots \vee D_m)$$

This restriction is motivated by the fact that it tightly corresponds to lists of keywords. The concepts C_i represent words from the shared terminology (compare section 5.2.2) one of which should occur in the description, the concepts D_j correspond to words from the shared terminology that should not occur in the description. Our experience is that this conception is quite natural for persons that normally use conventional search engines.

Set of basic terms: The concept names occurring the definition of the filler type may not be chosen arbitrarily, but they have to be taken from the set of concepts that fall under the domain restriction of the corresponding relation and they have to be from the shared ontology. This restriction serves two purposes. First of all, it facilitates information filtering across different classification hierarchies by rewriting as described in chapter 7. Second, it ensures that the concepts correspond to a word with a special meaning in the domain, because all concepts from the shared ontology

are words from the shared terminology (section 5.2.2)that in turn is built using domain specific thesauri (compare section 5.3).

In order to process queries posted by a user the reasoning module has to manage and incorporate different theories. First of all the descriptions of the different available information sources have to be encoded in a terminological language. Each information source is modeled as a sub-concept of the general concept 'information source'. It is further refined using relations from a shared ontology (compare definition 5.2). As mentioned above, this strategy leads to a good retrieval result.

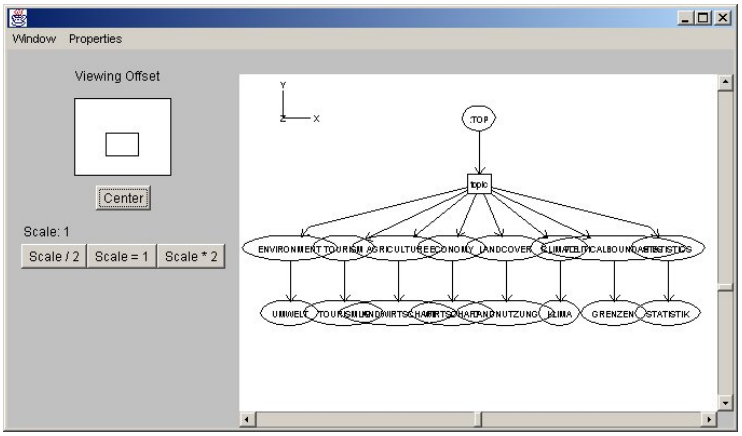


Figure 8.4: Part of the predefined query vocabulary

In order to enable the reasoner to compare the descriptions of information sources with the user query, the different background theories that constitute the shared ontology are also included in the reasoning process. Figure 8.4 shows a small part of a background model representing a hierarchy of topics of different information systems that can be found on the World Wide Web. In the same way models for the other relevant properties of an information source are included. In the current state of the system, we use subject, creator and coverage as criteria for retrieval.

8.3 Semantic Translation

In BUSTER the re-classification approach described in this thesis (section 7.2.3) is not mainly used for information filtering, but rather for integrating the content of retrieved information sources. Once a certain information source has been retrieved and considered useful, the user can define the process needed in order to convert the retrieved information into the format and structure required by his software environment or choose a predefined

translation process. In the BUSTER system, a translation process consists of a set of translation components which may be wrapper, mediators or other kinds of transformation programs. In order to define the process, the user has to specify the control flow between these components. The specification is stored in XML format and can be re-used in later translation processes.

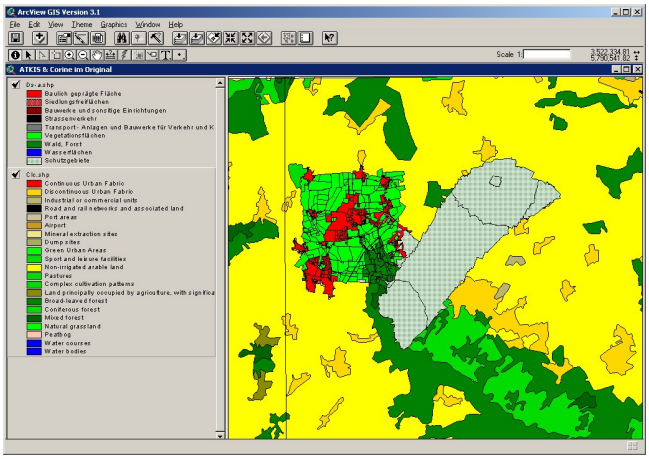


Figure 8.5: Two geographic data-sets before the semantic integration step

We exemplify a translation process using the problem of integrating ATKIS and CORINE data introduced in section 5.3.1. The first source `CORINE` stores its data in two tables[1]. The first table is called `clc_ns2`. Every entry represents one geological item. `clc_ns2` contains the attributes `CLC_NS2_ID` (identifier), `AREA` (surface in *ha*), and `NS` (classification). Especially the last attribute `NS` refers to catalogue, wherein all items are classified. In `CORINE`, the catalogue contains more than 64 concepts. The second table `clc_ns2_pol` stores polygons describing the area of an item. The attributes are `CLC_NS_ID` (reference to `clc_ns2`), `VERT_ID` (identifier of a vertices), `RWERT` (x–coordinate of the vertices), `HWERT` (y–coordinate of the vertices), and `NEXT_V_ID` (identifier of the following vertices). In the second source `ATKIS` a geological item is stored in one table `atkisf` with the attributes `id`, `rechts` (x–coordinate of the first vertice), `hoch` (y–coordinate of the first vertices), `fl` (surface in m^2), and `folie` (classification). Analogously to `CORINE` the last attribute `folie` refers to a classification catalogue containing more than 250 terms. But the catalogues of `CORINE` and `ATKIS` are different.

Obviously, the first task is to integrate the different data structures of the two information sources. In order to perform this task, BUSTER mainly uses

[1]For readability reasons the tables of both sources are simplified. Some attributes are omitted.

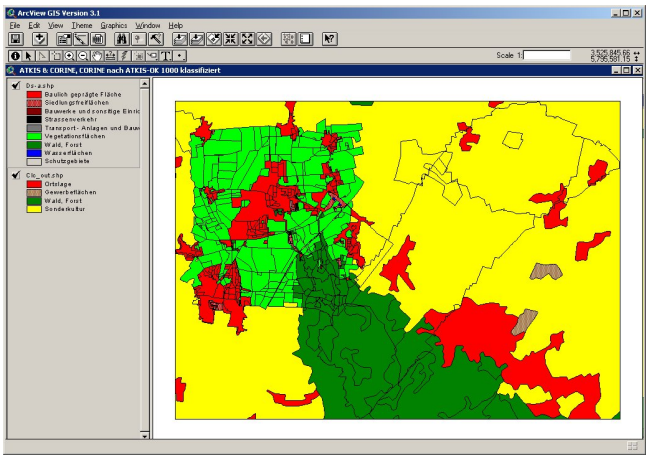


Figure 8.6: The data-sets after integration by re-classifcation

functionality provided by the MECOTA system [Wache, 1999], a rule-based mediator that can be considered the predecessor of the BUSTER system. MECOTA implements a classical mediator-wrapper architecture. Following this architecture, the CORINE information source is first converted into a standard Syntax using an ODBC Wrapper. In a second step, a commercial translation and manipulation system for geographic data is used in order to transform the spatial coordination system and select the region, we are interested in. After this transformation we get the situation displayed in figure 8.5: The two data-sets can be viewed as different layers in a common map.

At this point of the process a semantic translation step is needed, because the two data-sets are still classified according to different catalogue systems. In order to transform the CORINE classification into the ATKIS scheme, the BUSTER system exactly performs the re-classification described in section 7.2.3. The translation is performed by the MECOTA mediator, which already has the capability to convert entries in databases according to specific translation rules that have to be specified in advance. BUSTER uses this capability in a two step translation process. First, the reasoning module is called and performs the semantic re-classification. The result of this classification is encoded as a set of translation rules [Wache and Stuckenschmidt, 2001]. MECOTA uses these rules in order to actually execute the translation on the data. The result of this semantic translation step is shown in figure 8.6. As a last step, the now completely translated information is written into a new database using a second wrapper.

Conclusions

The BUSTER system that has been developed at the University of Bremen during the last two year provides an intelligent middle-ware for information sharing. It covers the problem of identifying and integrating heterogeneous and distributed information sources of related topics. The integration problem is addressed on different levels including syntax, structure and semantic integration. We successfully applied the system to integration problems from the area of geographic information processing. A second application in the domain of electronic commerce is currently being carried out. These applications provide a string evidence that the system can be used in real-life environments.

With respect to this thesis, the BUSTER system shows that the models and methods developed are not only of theoretical interest, but that they contribute to a practical solution for information sharing problems. In especially, we conclude that the general framework described in this thesis can be put to work using existing Web technologies: shared ontologies can be encoded in RDF schema, DAML+OIL can be used to build source ontologies and transformations between DAML+OIL and the more flexible framework DLML can be implemented using XSLT. Information sources in terms of collections of HTML documents can be classified using the WebMaster system. The generated classification metadata can be integrated into a standard metadata set assigned to every page. Finally, mapping and filtering methods can be implemented on top of existing subsumption reasoners that can be accessed over the Web. This tight coupling with existing technologies makes us optimistic about the potential contribution of the framework to a more intelligent *Semantic Web*.

Chapter 9

Discussion

The thesis defended in this work was that ontologies support information sharing also in weakly structured environments. In the previous chapters we discussed different issues connected to this claim and proposed a framework for using ontologies for information sharing on the Web. We now sum up our results and relate them to the global claim. We give an overview over the framework we developed and explicitly state our results and their contributions. We summarize with a discussion of two specific directions for future research implied by some of our results.

In this thesis we discussed the use on ontologies for information sharing in heterogeneous and weakly structured environments. We argued that such environments require new methods for representing, deploying and using ontologies as many results from previous work in the area of distributed databases do not directly apply. The discussion of requirements and solutions was guided by the following four question stated in the introduction:

A: How do we represent and reason about ontologies ?

B: What kind of ontologies are needed and how to build them ?

C: How can we connect ontologies and information ?

D: How can we use ontologies for improving semantic interoperability ?

We addressed question A in part I, questions B and C in part II and question D in part III of this thesis. Reviewing the results presented on these questions we see that the models and methods introduced are tightly related forming a framework for ontology-based information sharing in heterogeneous and weakly structured environments. Figure 9.1 gives an overview of this framework.

The figure illustrates that our framework consists of three main parts. The central one is a representational infrastructure defining how information,

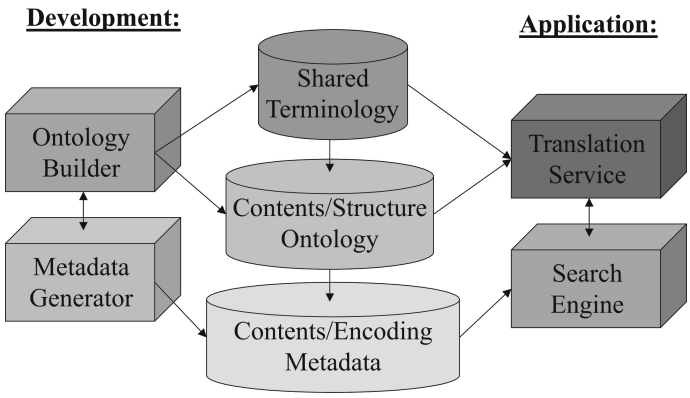


Figure 9.1: A framework for ontology-based information sharing

metadata, global- and local ontologies are represented (compare chapter 3) and arranged (compare chapter 5). The second part is concerned with tools to build this infrastructure, i.e. methods to engineer global and local ontologies (compare chapter 5) and to generate and manage ontology-based metadata (compare chapter 6). The third part is concerned with methods for translating and filtering information (compare chapter 7).

9.1 Conclusions

The overall conclusion of this work is that ontologies are indeed a suitable technology to support information sharing in weakly structured environments under certain conditions. We now discuss these conditions in more details.

9.1.1 Representational Infrastructure

The first condition for a successful application of ontologies to information sharing is the existence of a representational infrastructure. This infrastructure has to ensure that the potential benefits of ontologies can be exploited.

Semantics and Reasoning On a logical level, the representational infrastructure has to have a clear logical semantics that supports reasoning about contextual information. Having reasoning support is important, because the possibility to reason about the meaning of information is one of the major benefits of using ontologies in systems and not only for the communication between people. Semantics and reasoning can be used at ontology development time for consistency checking or at run-time for classifying individuals or performing semantic mappings. In our approach, the semantics of ontologies is provided by the general framework of Baader and others [Baader et al., 2000] that is general

enough to cover a wide range of ontology languages but still concrete enough to clearly define terminological reasoning, which is the basis for many results of the thesis.

Shared and Contextual Meaning On the content level, the benefits of ontologies can either be that they define a common understanding of specific terms. Thus, the reference to a term in such a shared ontology makes it possible to communicate between systems on a semantic level. On the other hand ontologies can be used to explicate background knowledge by defining what a certain term means in the context of the corresponding source. In order to make use of both of these benefits of ontologies the representational infrastructure has to consist of a mixture of shared and non-shared ontologies. In our framework non-shared ontologies define the meaning of classes in a specific information source using only terms from a shared ontology. This ensures that we can capture the contextual information of every information source but still have a basic understanding of terms that is shared between systems.

Content Metadata In order to be useful definitions of contextual interpretations have to be directly connected with concrete information, because we use ontologies as a tool for information sharing. The connection has to be tight enough to make the transfer of reasoning results from the logical level to the information. On the other hand the connection must be flexible enough to be applicable to weakly structured information without having to build these structures from the scratch. In our framework, we use an assignment of complete Web pages to ontological classes with RDF schema metadata. This assignment is easy to establish by means of machine learning techniques and it supports many useful methods like validation, querying and content based browsing of Web resources.

9.1.2 Infrastructure Development

Another important point is that it is not sufficient to define an infrastructure for information sharing. Providing methodologies and tools for supporting the development of this infrastructure is at least equally important since knowledge acquisition is well known to be one of the main bottlenecks in the application of knowledge-based technology.

Ontology Construction The success of ontology-based information sharing heavily depends on the quality of the ontologies used. Building them in an ad hoc way leads to serious shortcomings. In [Stuckenschmidt et al., 2000] we describe problems that occurred in an early case study on semantic matching that were mostly caused by sloppy ontology development. It clearly displayed the need for modeling guidance. Existing ontology engineering approaches are often very general, they state general principles but only provide limited support for a more concrete modeling task. We found out that for the case of building source

ontologies and shared vocabularies, a bottom-up approach is suitable that takes the actual integration problem as a starting point and consults general models like top level ontologies and linguistic resources only if necessary. The resulting vocabularies are general enough to cover at least a certain class of integration problems. We think that this is more valuable than a general top-down approach because it solves real world problems without losing the connection to basic ontological principles.

Metadata Generation As metadata plays an important role especially in weakly structured environments. At the same time, the creation of such metadata is harder the less structure is present to refer to. Despite these problems we think that successful approaches to applying ontologies in these environments will have to live with the existing structures, in our case HTML documents, because the freedom from the need to encode sophisticated data structures is one of the secrets behind the success of the World Wide Web. We therefore proposed an approach for mostly automatically generating metadata that links information to ontologies. We claim that the assignment of individual Web pages to classes in an ontology provides a good trade-off between the strength of the connection and the effort of establishing it. We show that Web-page classification can be done using classification rules that refer to the structure of HTML documents. The resulting classification can be used for content-based navigation and search. We also demonstrated that structural classification rules can be generated in a mostly automatic way using techniques from machine learning.

9.1.3 Using the Infrastructure

The final important condition for the successful application of ontologies for information sharing is that the approach chosen scales up to real-life problems. This claim raises new questions with respect to compatibility with existing technology and with tolerance for imperfect data and knowledge.

Compatibility Solutions developed in science often fail to make their way into real applications due to a lack of compatibility with industrial standards. Considering the World Wide Web as a target application area, the World Wide Web Consortium provides a platform where science and industry make an effort for the development of joint standards. We therefore think that any approach to ontology-based information sharing on the World Wide Web should be compatible with W3C standards. The general framework described in this thesis can be put to work using existing Web technologies: shared ontologies can be encoded in RDF schema, DAML+OIL can be used to build source ontologies and transformations between DAML+OIL and the more flexible framework DLML can be implemented using XSLT. Information sources in terms of collections of HTML documents can be classified using the WebMaster system. Finally, mapping and filtering methods can be implemented on top of existing subsumption reasoners that can be accessed over the Web.

Robust Methods Unlike conventional approaches to knowledge representation and reasoning, the application of ontologies in weakly structured and heterogeneous environments can in principle make no assumption about the quality of the information that has to be handled. As argued above, we can provide methodological guidance for the development of ontologies and the generation of metadata, but we still may have to face inconsistencies or incompleteness when trying to perform reasoning across different systems. We claim that there is a need for developing new reasoning techniques that are more flexible and fault tolerant than classical deduction systems in order to cope with the nature of the application environment. The approximate mapping approach proposed in chapter 7 is an example of such a more flexible method. We can answer Boolean queries over concept names by replacing unknown concept names by their lower bound in the corresponding source ontology. The resulting query can be processed in the context of the remote information source delivering a query result that can be proven to be a correct approximation of the intended result.

9.2 Future Research

Obviously, there are many possible directions for further research due to the importance and difficulty of the problem addressed. Listing them all would be too much in this context. We will rather pick out two topics that we assume to be fundamental for knowledge representation and reasoning on the Web, namely investigation of representational patterns and terminological reasoning based on approximations.

9.2.1 Representational Patterns

The second part of this thesis was concerned with a general logical framework for representing and reasoning about terminological knowledge. All considerations made were based on the notion of logical operators, how they can be used to model concepts and how we can translate between them in order to keep some formal properties. While the discussion on the level of operators is convenient for logical investigations, it seems to be inappropriate from a modeling perspective. Experiences from software engineering have shown that the notion of a pattern is very useful in order to support the creation and exchange of models, because they provide the right level of abstraction [Williamson et al., 2000]. Recently, a discussion about patterns also started in the knowledge acquisition and modeling community (see [Menzies, 1997, Reich, 1999, Puppe, 2000, Devedzic, 1999, Clark et al., 2000, Staab and Maedche, 2000]). We think that the notion of patterns in knowledge modeling should be investigated in more detail, because such patterns can have the following benefits for ontology-based information sharing and knowledge processing in general.

Guidance for Ontology Modeling One of the most often cited advantages of patterns is the guidance they provide for people in the sense that one does not have to start from the scratch in order to find the solution to a given problem. Patterns rather support a middle-out approach to problem solving by providing building blocks to start with. The process of solving the problem then consists of finding the right combination of patterns and adapting and refining them to fit together and to meet the goal. These advantages also hold for ontology patterns. The pattern defines a structure over different modeling primitives. This releases the user from the problem of finding the right primitives and a way of how to combine them. He can rather concentrate on the adaptation of the structure to his special needs (e.g. by assigning domain-specific names to modeling primitives in the structure). This will be a great advantage considering the World Wide Web where we are concerned with people that are not skilled in developing ontologies.

Validation and Verification The notion of an ontology pattern does not only come with a structure providing guidance for the process of building the ontology. It also provides means to validate an ontology that has been build. The validation even includes formal verification based upon the constraints defined in the pattern. The possibility of verifying is a major advantage for example over software design patterns, which often only consist of a class diagram with natural language explanations. These patterns also provide guidance, however they fail to capture whether a system really implements the patterns or not. For the case of computer systems this is no problem as long as the system fulfills its requirements. In the case of ontologies for the World Wide Web we have to deal with the situation that intelligent agents will make automatic use of ontologies without user interference. If we can guarantee that an ontology is consistent with a certain pattern, this would be a great advantage for intelligent agents because they only have to check whether they understand the pattern in order to decide whether they can make use of an ontology specification

Ontology Libraries and Reuse Originally, ontologies were introduce into the field of artificial intelligence in order to support large scale knowledge sharing and reuse. While some progress could be made in re-using top-level ontologies concerned with very abstract concepts it soon turned out that the re-use of domains comes with many unsolved problems not to mention the reuse of application ontologies. It turned out that almost all lower level ontology specifications are strongly biased by their intended use making it difficult to use them for a different purpose. Rather than perceiving it as a problem, patterns take advantage of the fact that different purposes demand for different conceptualizations. By making the impact of the purpose explicit in the ontology patterns we can support re-use of domain ontologies. Libraries of ontologies could be organized on the basis of different patterns. This would enable a user to decide whether an ontology is likely to be designed in a way that supports his own purpose or if major changes would be required to re-use the ontology. This

could help to push the border of re-usability of ontologies further down from top-level ontologies towards domain and application ontologies.

Mapping, Merging and Translation Mapping, merging and translation of ontologies is an important topic especially for the vision of an intelligent Web, where different users that do not even know of each other are expected to define their own ontologies in order to describe their Web-site. In order to enable intelligent agents to reason across these ontologies they have to be integrated using one of the above mentioned approaches. The modeling bias as well as the use of different languages almost makes this a mission impossible unless there is a common basis to agree upon. While ontology patterns support the heterogeneity of ontology specifications they still preserve a common basis in terms of the basic modeling primitives used. A pattern may or may not use a modeling primitive, but if a certain primitive occurs in two patterns we can assume that they are the same on the conceptual level. This conceptual correspondence provides a basis for the definition of mappings and translations between ontologies. On a higher level, the same holds for ontology specifications using the same pattern. In this case we can also assume that the basic structures of the specifications are the same. This helps us to merge ontologies in a consistent way.

9.2.2 Approximate Terminological Reasoning

Closely related to the question of mapping, merging and translation is the use of approximate reasoning techniques. In chapter 7 we presented a specific method for approximating class definitions and concept queries based on the work of Selman and Kautz [Selman and Kautz, 1996]. Beyond this work, there is significant work on approximate reasoning in a logical setting that could be used in terminological reasoning as well.

Approximation Techniques

Knowledge Compilation In order to avoid complexity at run-time, knowledge compilation [Darwiche and Marquis, 2001] aims at explicating knowledge hidden in a logical model in a pre-processing step. Derived facts are added to the original theory as axioms avoiding the need to deduct them again. In the case of ontological reasoning, implicit subsumption and membership relations are good candidates for a compilation. For example, implicit subsumption relations in a DAML+OIL ontology could be identified using a description logic reasoner, the resulting more complete hierarchy could be encoded in RDF schema and used by systems that do not have the ability to perform complex reasoning. Approximations come into play, because the compiled model is not always complete.

Language Weakening The idea of language weakening is based on the well-known trade-off between the expressiveness and the reasoning complexity of a logical language. By weakening the logical language a theory is encoded in, we are able to tread the completeness of reasoning against run-time. The logic

that underlies DAML+OIL for example is known to be highly intractable, existing reasoners therefore use a slightly weaker logic that still allows to compute most deductions. This idea can be further extended by starting with a very simple language and iterating over logics of increasing strength supplementing previously derived facts.

Approximate Deduction Instead of modifying the logical language, approximations can also be achieved by weakening the notion of logical consequence [Schaerf and Cadoli, 1995],[Dalal, 1996]. This can be done by restricting the size of definitions that are considered or by allowing failures on parts of the vocabulary used. On the semantic Web, using the latter techniques will often even be necessary if agents that only share parts of their vocabulary. In this case approximate deduction can be used to compute at least those deductions that solely depend on the shared vocabulary instead of failing to derive anything.

Expected Benefits

We think that applying this work to the problem of terminological reasoning in heterogeneous environments in a more systematic way is a very promising research area for the below mentioned reasons:

Interoperability Approximate terminological reasoning can help to compare, align and integrate heterogeneous ontologies in complex application scenarios. When sharing or re-using ontologies that are not designed to be used together, we will often face the situation where classes describe similar, but not equivalent sets of objects or they will structure the domain according to orthogonal dimensions. In these cases, we need approximate reasoning techniques in order to identify similar concepts and to relate heterogeneous concepts.

We already exploited this in developing translation and information filtering methods in chapter 7. However, there are more possibilities for getting better approximations. Further, the methods developed in this thesis have to be extended to cover other application areas. A very interesting connection is the one to communication in multi agent systems, where the agents use both, shared and non-shared ontologies.

Robustness Approximate terminological reasoning can be used to cope with erroneous, inconsistent or missing information for example by restricting the vocabulary of an application domain to a maximal handable subset. This ability will be very important if we assume a large scale use of ontologies on the semantic Web that will be dominated by sloppy ontologies that claim neither completeness nor consistency.

Another important problem where robust reasoning is required is the ability to cope with changing ontologies. Information sources may be annotated with

terms from an older version of an ontology. The new version may contain previously non-existent terms and new or revised definitions. The situation becomes even more complicated when terms that are used to describe information are removed from an ontology. In this case, approximation may help to make at least approximate statements about the information.

Scalability Approximate terminological reasoning can be used to reduce complex reasoning tasks to simpler ones e.g. by choosing a less expressive encoding language. Using different simplifications with increasing exactness, we can produce an anytime behaviour of terminological reasoning engines. While abandoning completeness, such simplifications will help to scale up logical reasoning to realistic application domains on the World Wide Web.

A first step towards using approximations for scalability has already been made in this thesis: In chapter 4 we discussed transformations that weaken a theory. However, in order to use these kinds of approximations to scale up reasoning, we have to investigate the expressiveness of logical languages in a more systematic way and to analyze the loss of information implied by a weakening.

Bibliography

[Aben, 1993] Aben, M. (1993). Formally specifying re-usable knowledge model components. *Knowledge Acquisition Journal*, 5:119–141.

[AdV, 1998] AdV (1998). Amtliches Topographisch Kartographisches Informationssystem ATKIS. Technical report, Landesvermessungsamt NRW, Bonn.

[Arens et al., 1993] Arens, Y., Chee, C., Hsu, C.-N., and Knoblock, C. (1993). Retrieving and integrating data from multiple information sources. *International Journal of Intelligent and Cooperative Information Systems*, 2(2):127–158.

[Baader et al., 2000] Baader, F., Lutz, C., Sturm, H., and Wolter, F. (2000). Fusions of description logics. In *Proceedings of the International Workshop in Description Logics 2000 (DL2000)*, volume 33 of *CEUR Workshop Proceedings*.

[Baader et al., 2002] Baader, F., Lutz, C., Sturm, H., and Wolter, F. (2002). Fusions of description logics and abstract description systems. *Journal of Artificial Intelligence Research*, 16:1 – 58.

[Basili et al., 2001] Basili, R., Moschitti, A., and Pazienza, M. T. (2001). NLP-driven IR: Evaluating performances over a text classification task. In Nebel, B., editor, *Proceedings of the 13th International Joint Conference on Artificial Intelligence IJCAI-01*, pages 1286–1294.

[Bechhofer et al., 1999] Bechhofer, S., Horrocks, I., Patel-Schneider, P. F., and Tessaris, S. (1999). A proposal for a description logic interface. In *Proceedings of the Description Logic Workshop DL'99*, pages 33–36.

[Bechofer et al., 2001] Bechofer, S., Horrocks, I., Goble, C., and Stevens, R. (2001). OilEd: A reason-able ontology editor for the semantic web. In Baader, F., Brewka, G., and Eiter, T., editors, *KI 2001: Advances in Artificial Intelligence*, pages 396–408. Springer.

[Belkin and Croft, 1992] Belkin, N. and Croft, B. (1992). Information filtering and information retrieval: Two sides of the same coin? *Communications of the ACM*, 35(12):29–38.

[Benjamins and Fensel, 1998] Benjamins, V. and Fensel, D. (1998). The ontological engineering innitiative $(KA)^2$. In Guarino, N., editor, *Proceedings of the International Conference on Formal Ontologies in Information Systems (FOIS-98)*, Frontiers in Artificial Intelligence and Applications, pages 287–301. IOS-Press, Trento, Italy.

[Bergamaschi et al., 1999] Bergamaschi, S., Castano, S., and Vincini, M. (1999). Semantic integration of semi-structured and structured data sources. *SIGMOD Records*, 28(1):54–59.

[Biron and Malhotra, 2000] Biron, P. and Malhotra, A. (2000). XML schema part 2: Datatypes. Working draft, W3C. http://www.w3.org/TR/2000/WD-xmlschema-2-20000225/.

[Boley et al., 1999] Boley, D., Gini, M., Gross, R., Han, E.-H. S., Hastings, K., Karypis, G., Kumar, V., Mobasher, B., , and Moor, J. (1999). Document categorization and query generation on the world wide web using webace. *AI Review*, 13(5-6):365–391.

[Brachman, 1977] Brachman, R. (1977). What's in a concept: Structural foundations for semantic nets. *International Journal of Man-Machine Studies*, 9(2):127–152.

[Bray et al., 1998] Bray, T., Paoli, J., and Sperberg-McQueen, C. (1998). Extensible markup language (xml) 1.0. Technical Report REC-xml, W3C. http://www.w3.org/TR/REC-xml.

[Brickley et al., 1998] Brickley, D., Guha, R., and Layman, A. (1998). Resource description framework (RDF) schema specification. Working draft, W3C. http://www.w3c.org/TR/WD-rdf-schema.

[Calvanese et al., 1998a] Calvanese, D., Giacomo, G. D., and Lenzerini, M. (1998a). On the decidability of query containment under constraints. In *Proc. 17th ACM Symposium on Principles of Database Systems*, pages 149–158.

[Calvanese et al., 2001] Calvanese, D., Giacomo, G. D., and Lenzerini, M. (2001). A framework for ontology integration. In *Proceedings of the international semantic web working symposium*, pages 303–316, Stanford, USA.

[Calvanese et al., 1998b] Calvanese, D., Giacomo, G. D., Lenzerini, M., Nardi, D., and Rosati, R. (1998b). Description logic framework for information integration. In *Proceedings of the Internaltional Conference on Principles of Knowledge Representation and Reasoning, KR-98*, pages 2–13.

[Calvanese et al., 1999] Calvanese, D., Lenzerini, M., and Nardi, D. (1999). Unifying class-based representation formalisms. *Journal of Artificial Intelligence Research*, 11:199–240.

[Chalupsky, 2000] Chalupsky, H. (2000). Ontomorph: A translation system for symbolic logic. In [Cohn et al., 2000], pages 471–482.

[Champin, 2000] Champin, P.-A. (2000). RDF tutorial. Available at http://www710.univ-lyon1.fr/ champin/rdf-tutorial/.

[Chang and Garcia-Molina, 2001] Chang, K.-C. and Garcia-Molina, H. (2001). Approximate query mapping: Accounting for translation closeness. *The VLDB Journal*, 10:155–181.

[Chen, 1976] Chen, P.-S. (1976). The entity relationship model - towards a unified view of data. *ACM Transactions on Database Systems*, 1(1):9–36.

[Ciocoiu and Nau, 2000] Ciocoiu, M. and Nau, D. S. (2000). Ontology-based semantics. In [Cohn et al., 2000], pages 539–546.

[Clark et al., 2000] Clark, P., Thompson, J., and Porter, B. (2000). Knowledge patterns. In [Cohn et al., 2000], pages 591–600.

[Cohn et al., 2000] Cohn, A. G., Giunchiglia, F., and Selman, B., editors (2000). *KR2000: Principles of Knowledge Representation and Reasoning*. Morgan Kaufmann, San Francisco.

[Craven et al., 2000] Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., and Slattery, S. (2000). Learning to construct knowledge bases from the world wide web. *Artificial Intelligence*, 118(1-2):69–113.

[Dalal, 1996] Dalal, M. (1996). Semantics of an anytime family of reasoners. In *Proceedings if the 12th European Conference on Artificial Intelligence*, pages 360–364.

[Darwiche and Marquis, 2001] Darwiche, A. and Marquis, P. (2001). A perspective on knowledge compilation. In *Proceedings of the International Joint Conference on Artificial Intelligence IJCAI-01*, pages 175–182.

[De Giacomo, 1995] De Giacomo, G. (1995). *Decidability of Class-Based Knowledge Representation Formalisms*. Phd thesis, University Degli Studi Di Roma "La Sapienze".

[Decker et al., 2000] Decker, S., Melnik, S., Harmelen, F. V., Fensel, D., Klein, M., Broekstra, J., Erdmann, M., and Horrocks, I. (2000). The semantic web: The roles of XML and RDF. *IEEE Expert*, 15(3):63–74.

[Deerwester et al., 1990] Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., and Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407.

[Devedzic, 1999] Devedzic, V. (1999). Ontologies: borrowing from software patterns. *ACM Intelligence Magazine*, 10(3):14–24.

[Donini et al., 1991] Donini, F., Lenzerini, M., Nardi, D., and Nutt, W. (1991). The complexity of concept languages. In Sandewall, J. A., Fikes, R., and E., editors, *2nd International Conference on Knowledge Representation and Reasoning, KR-91*, pages 151–162. Morgan Kaufmann.

[Donini et al., 1996] Donini, F. M., Lenzerini, M., Nardi, D., , and Schaerf, A. (1996). Reasoning in description logics. In Brewka, G., editor, *Principles of Knowledge Representation*, Studies in Logic, Language and Information, pages 193–238. CSLI Publications.

[Duineveld et al., 1999] Duineveld, A., Stoter, R., Weiden, M., Kenepa, B., and Benjamins, V. (1999). Wondertools? a comparative study of ontological engineering tools. In [Gaines et al., 1999].

[European Environmental Agency, 1999a] European Environmental Agency (1997-1999a). Corine land cover. technical guide. Technical report, European Environmental Agency. ETC/LC, European Topic Centre on Land Cover.

[European Environmental Agency, 1999b] European Environmental Agency (1999b). GEMET - general multilingual environmental thesaurus. Technical report, European Topic Centre on Catalogue of Data Sources (ETC/CDS) European Environmental Agency. Version 2.0.

[Euzenat, 2001] Euzenat, J. (2001). An infrastructure for formally ensuring interoperability in a heterogeneous semantic web. In Cruz, I. F., Decker, S., Euzenat, J., and McGuinness, D., editors, *Proceedings of the First Semantic Web Working Symposium*, pages 345–360, Stanford, CA, USA.

[Euzenat, 2001] Euzenat, J. (2001). Preserving modularity in XML encoding of description logics. In McGuinness, D., Patel-Schneider, P., Goble, C., and Moeller, R., editors, *Proc. 14th workshop on description logics (DL), Stanford (CA US)*, pages 20–29.

[Euzenat and Stuckenschmidt, 2003] Euzenat, J. and Stuckenschmidt, H. (2003). The 'family of languages' approach to semantic interoperability. In Omelayenko, B. and Klein, M., editors, *Knowledge Transformation for the Semantic Web*, Amsterdam. IOS Press.

[Fallside, 2000] Fallside, D. C. (2000). XML schema part 0: Primer. Working draft, W3C. http://www.w3.org/TR/2000/WD-xmlschema-0-20000225/.

[Farquhar and Gruninger, 1997] Farquhar, A. and Gruninger, M. (1997). Proceedings of the AAAI spring symposium on ontological engineering. Technical report, AAAI, Stanford, CA.

[Fellbaum, 1998] Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database*. Language, Speech, and Communication Series. MIT Press.

[Fensel, 1999] Fensel, D., editor (1999). *Proceedings of the IJCAI -99 Workshop on Intelligent Information Integration*, volume 23 of *CEUR Workshop Proceedings*. CEUR Publications and AIFB Karlsruhe.

[Fensel, 2001] Fensel, D. (2001). *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*. Springer-Verlag, Berlin.

[Fensel et al., 1998] Fensel, D., Decker, S., Erdmann, M., and Studer, R. (1998). Ontobroker: The very high idea. In *11. International Flairs Conference (FLAIRS-98)*, pages 131–135, Sanibal Island, USA.

[Fensel et al., 1997] Fensel, D., Erdmann, M., and Studer, R. (1997). Ontology groups: Semantically enriched subnets of the WWW. In *Proceedings of the International Workshop Intelligent Information Integration during the 21st German Annual Conference on Artificial Intelligence*, Freiburg, Germany.

[Fensel et al., 2000] Fensel, D., Horrocks, I., Harmelen, F. V., Decker, S., Erdmann, M., and Klein, M. (2000). OIL in a nutshell. In *Proceedings of the 12th International Conference on Knowledge Engineering and Knowledge Management EKAW 2000*, pages 1–16, Juan-les-Pins, France.

[Fensel et al., 2001] Fensel, D., Horrocks, I., van Harmelen, F., McGuinness, D. L., and Patel-Schneider, P. F. (2001). OIL: An ontology infrastructure for the semantic web. *IEEE Intelligent Systems*, 16(2):38–44.

[Frakes and Baeza-Yates, 1992] Frakes, W. B. and Baeza-Yates, R. (1992). *Information Retrieval: Data Structures and Algorithms*. Prentice-HALL, North Virginia.

[Freitag and Kushmerick, 2000] Freitag, D. and Kushmerick, N. (2000). Boosted wrapper induction. In *Proceedings of AAAI-00*, pages 577–583, Austin, TX.

[Gaines et al., 1999] Gaines, B., Kremer, R., and Musen, M. (1999). Proceedings of the 12th Banff knowledge acquisition for knowledge-based systems workshop. Technical report, University of Calgary/Stanford University.

[Gaizauskas and Humphreys, 1997] Gaizauskas, R. and Humphreys, K. (1997). Using a semantic network for information extraction. *Journal of Natural Language Engineering*, 3(2/3):147–169.

[Garcia-Molina et al., 1997] Garcia-Molina, H., Papakonstantinou, Y., Quass, D., Rajaraman, A., Sagiv, Y., and Ullman, J. (1997). The TSIMMIS project: Integration of heterogeneous information sources. *Journal of Intelligent Information Systems*, 8(2):117–132.

[Garcia-Molina et al., 1995] Garcia-Molina, H., Papakonstantinou, Y., Quass, D., Rajaraman, A., Sagiv, Y., Ullman, J., and Widom, J. (1995). The TSIMMIS approach to mediation: Data models and languages. In *Next Generation Information Technologies and Systems (NGITS-95)*, Naharia, Israel. Extended Abstract.

[Genesereth and Fikes, 1992] Genesereth, M. and Fikes, R. (1992). Knowledge interchange format version 3.0 reference manual. Report of the Knowledge Systems Laboratory KSL 91-1, Stanford University.

[Goasdoue and Reynaud, 1999] Goasdoue, F. and Reynaud, C. (1999). Modeling information sources for information integration. In Fensel, D. and Studer, R., editors, *Knowledge Acquisition, Modeling and Management*, Lecture Notes in Artificial Intelligence, pages 121–138, Berlin. Springer.

[Goh et al., 1994] Goh, C., Madnick, S., and Siegel, M. (1994). Context interchange: overcoming the challenges of large-scale interoperable database systems in a dynamic environment. In *Proceedings of the Third International Conference on Information and Knowledge Management*, pages 337–346.

[Gomez-Perez and Corcho, 2002] Gomez-Perez, A. and Corcho, O. (2002). Ontology langauges for the semantic web. *IEEE Intelligent Systems*, January/February:54–60.

[Gomez-Perez et al., 2001] Gomez-Perez, A., Gruninger, M., Stuckenschmidt, H., and Uschold, M. (2001). Proceedings of the IJCAI-01 workshop on ontologies and information sharing. Technical Report 47, CEUR workshop proceedings, Seattle, USA.

[Gomez-Perez and Juristo, 1997] Gomez-Perez, M. F. A. and Juristo, N. (1997). Methontology: From ontological arts towards ontological engineering. In [Farquhar and Gruninger, 1997], pages 33–40.

[Gruber, 1991] Gruber, T. (1991). Ontolingua: A mechanism to support portable ontologies. KSL Report KSL-91-66, Stanford University.

[Gruber, 1993] Gruber, T. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220.

[Gruninger and Uschold, 2002] Gruninger, M. and Uschold, M. (2002). Ontologies and semantic integration. Part of a Report to the US Government on State of the Art in Agent Technology.

[Guarino and Giaretta, 1995] Guarino, N. and Giaretta, P. (1995). Ontologies and knowledge bases: Towards a terminological clarification. In Mars, N., editor, *Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing*, pages 25–32. IOS Press, Amsterdam.

[Guarino et al., 1999] Guarino, N., Masolo, C., and Vetere, G. (1999). Ontoseek: Content-based access to the web. *IEEE Intelligent Systems*, 14(3):70–80.

[Haarslev and Moller, 2001] Haarslev, V. and Moller, R. (2001). Description of the RACER system and its applications. In *Proceedings of the Description Logics Workshop DL-2001*, pages 132–142, Stanford, CA.

[Hartmann, 2002] Hartmann, J. (2002). Lernen struktureller Regeln zur Klassifikation von Web Dokumenten. Master's thesis, University of Bremen.

[Hartmann and Stuckenschmidt, 2002] Hartmann, J. and Stuckenschmidt, H. (2002). Automatic metadata analysis for environmental information systems. In *Proceedings of the International Symposium on Environmental Informatics*.

[Horrocks, 1998] Horrocks, I. (1998). The FaCT system. In de Swart, H., editor, *Automated Reasoning with Analytic Tableaux and Related Methods: International Conference Tableaux'98*, number 1397 in Lecture Notes in Artificial Intelligence, pages 307–312. Springer-Verlag, Berlin.

[Horrocks, 2001] Horrocks, I. (2001). Reasoning with daml+ oil: What can it do for you? Presentation given at DAML PI Meeting.

[Horrocks et al., 1999] Horrocks, I., Sattler, U., Tessaries, S., and Tobies, S. (1999). Query containment using a dlr abox - preliminary version. LCTS Report 99-15, RWTH Aachen.

[ISO-8879, 1986] ISO-8879 (1986). Information processing – text and office systems - standard generalized markup language (SGML). Standard by the International Organization for Standardization.

[Jasper and Uschold, 1999] Jasper, R. and Uschold, M. (1999). A framework for understanding and classifying ontology applications. In [Gaines et al., 1999].

[Jenkins et al., 1999] Jenkins, C., Jackson, M., Burdon, P., and Wallis, J. (1999). Automatic RDF metadata generation for resource discovery. *Computer Networks*, 31:1305–1320.

[Karp et al., 2002] Karp, P., Chaudri, V., and Thomere, J. (2002). An XML-based ontology exchange language. Available at `http://www.ai.sri.com/ pkarp/xol`.

[Kashyap and Sheth, 1998] Kashyap, V. and Sheth, A. (1998). Semantic heterogeneity in global information systems: The role of metadata, context and ontologies. In Papazoglou, M. P. and Schlageter, G., editors, *Cooperative Information Systems*, pages 139–178. Academic Press, San Diego.

[Kashyap and Sheth, 1996] Kashyap, V. and Sheth, A. P. (1996). Semantic and schematic similarities between database objects: A context-based approach. *VLDB Journal: Very Large Data Bases*, 5(4):276–304.

[Keinitz, 1999] Keinitz, S. (1999). Aggregation in der Datenintegration. Master's thesis, Department of Mathematics and Computer Science, University of Bremen.

[Kent, 2002] Kent, R. (2002). Conceptual knowledge modelling language. available at `http://www.ontologos.org/CKML/`.

[Kim and Seo, 1991] Kim, W. and Seo, J. (1991). Classifying schematic and data heterogeinity in multidatabase systems. *IEEE Computer*, 24(12):12–18.

[Klein, 2001] Klein, M. (2001). Combining and relating ontologies: an analysis of problems and solutions. In [Gomez-Perez et al., 2001].

[Kottmann, 1999] Kottmann, C. (1999). Semantics and information communities. OGC Abstract Specification Topic 14, The Open GIS Consortium.

[Kurtonina and de Rijke, 1999] Kurtonina, N. and de Rijke, M. (1999). Expressiveness of concept expressions in first-order description logics. *Artificial Intelligence*, 107(2):303–333.

[Lassila and Swick, 1999] Lassila, O. and Swick, R. (1999). Resource description framework (RDF). Proposed recommendation, W3C. http://www.w3c.org/TR/WD-rdf-syntax.

[Lenat, 1998] Lenat, D. (1998). The dimensions of context space. Available on the web-site of the Cycorp Corporation. (http://www.cyc.com/publications).

[Lenat, 1995] Lenat, D. B. (1995). Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11).

[Levy, 1999] Levy, A. (1999). Combining artificial intelligence and databases for data integration. In Wooldridge, M. and Veloso, M., editors, *Artificial Intelligence Today: Recent Trends and Developments*, volume 1600 of *Lecture Notes in Computer Science*, pages 249–268. Springer, Berlin.

[Levy et al., 1996] Levy, A., Rajaraman, A., and Ordille, J. J. (1996). Querying heterogeneous information sources using source descriptions. In *Proceedings of the 22nd International Conference on Very Large Databases, VLDB-96*, pages 251–262, Bombay, India.

[Lewis, 1996] Lewis, D. D. (1996). Natural language processing for information retrieval. *Communcations of the ACM*, 39(1):92–101.

[Luke and Hefflin, 2002] Luke, S. and Hefflin, J. (2002). SHOE 1.01 proposal specification. Available at `http://www.cs.umd.edu/projects/plus/SHOE`.

[Maynard and Ananiadou, 1998] Maynard, D. and Ananiadou, S. (1998). Term sense disambiguation using a domain-specific thesaurus. In *Proc. of 1st International Conference on Language Resources and Evaluation (LREC)*, Granada, Spain.

[Menzies, 1997] Menzies, T. (1997). Object-oriented patterns: Lessons from expert systems. *Software: Practice and Experience*, 27(12):1457–1478.

[Motta, 1999] Motta, E. (1999). *Reusable Components for Knowledge Models: Case Studies in Parametric Design Problem Solving*, volume 53 of *Frontiers in Artificial Intelligence and Applications*. IOS Press.

[Muggleton, 1995] Muggleton, S. (1995). Inverse entailment and Progol. *New Generation Computing, Special issue on Inductive Logic Programming*, 13(3-4):245–286.

[Muggleton, 1999] Muggleton, S. (1999). Inductive logic programming: issues, results and the LLL challenge. *Artificial Intelligence*, 114(1–2):283–296.

[Muggleton and de Raedt, 1994] Muggleton, S. and de Raedt, L. (1994). Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19,20:629–679.

[Neumann et al., 2001] Neumann, H., Schuster, G., Stuckenschmidt, H., Visser, U., Voegele, T., and Wache, H. (2001). Intelligent brokering of environmental information with the buster system. In Hilty, L. and Gilgen, P., editors, *Sustainability in the Information Society - 15th International Symposium Informatics for Environmental Protection*, pages 505–512. Metropolis.

[Patel-Schneider and Swartout, 1994] Patel-Schneider, P. and Swartout, B. (1994). Description logic knowledge representation system specification. AT&T bell laboratories report, KRSS group of the DARPA knowledge sharing effort, Murray Hill, NJ.

[Patil et al., 1991] Patil, R., Fikes, R., , Patel-Schneider, P., McKay, D., Finin, T., and andn R. Neches, T. G. (1991). The DARPA knowledge sharing effort: Progress report. In Rich, C., Nebel, B., and Swartout, W., editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference*, Cambridge,MA.

[Pepper and Moore, 2001] Pepper, S. and Moore, G. (2001). XML topic maps (XTM) 1.0. Xtm specification, topicmaps.org. http://www.topicmaps.org/xtm/1.0/.

[Pierre, 2001] Pierre, J. M. (2001). On the automated classification of web sites. *Electronic Transactions on Artificial Intelligence*, 6. http://www.ida.liu.se/ext/etai/ra/seweb/002/.

[Preece et al., 1999] Preece, A., Hui, K.-Y., Gray, W., Marti, P., Bench-Capon, T., Jones, D., and Cui, Z. (1999). KRAFT architecture for knowledge fusion and transformation. In *19th SGES International Conference on Knowledge-based Systems and Applied Artificial Intelligence (ES'99)*, Berlin. Springer.

[Puppe, 2000] Puppe, F. (2000). Knowledge formalization patterns. In *Proceedings of Pacific Rim Knowledge Acquisition Workshop the PKAW 2000*.

[Ragget et al., 1999] Ragget, D., Le Hors, A., and Jacobs, I. (1999). HTML 4.01 specification. Recommendation, W3C. http://www.w3.org/TR/1999/REC-html401-19991224.

[Reich, 1999] Reich, J. (1999). Ontological design patterns for the integration of molecular biological information. In *Proceedings of the German Conference on Bioinformatics GCB'99*, pages p.156–166, Hannover, Germany.

[Rousset, 1997] Rousset, M.-C. (1997). Verifying the world wide web: a position statement. In van Harmelen, F. and J. van Thienen, editors, *Pro-ceedings of the Fourth European Symposium on the Validation and Verification of Knowledge Based Systems (EUROVAV97)*.

[Rumbaugh et al., 1998] Rumbaugh, J., Jacobson, I., , and Booch, G. (1998). *The Unified Modeling Language Reference Manual*. Addison-Wesley.

[Salton, 1986] Salton, G. (1986). Another look at automatic text-retreival systems. *Communications of the ACM*, 29(7):648–656.

[Salton and McGill, 1983] Salton, G. and McGill, M. (1983). *Introduction to modern information retrieval*. McGraw-Hill.

[Schaerf, 1994] Schaerf, A. (1994). Reasoning with individuals in concept languages. *Data Knowledge Engineering*, 13(2):141–176.

[Schaerf and Cadoli, 1995] Schaerf, M. and Cadoli, M. (1995). Tractable reasoning via approximation. *Artificial Intelligence*, 74(2):249–310.

[Schreiber et al., 1994] Schreiber, A., Wielinga, B., Akkermans, H., van der Velde, W., and Anjewierden, A. (1994). CML the CommonKADS conceptual modeling language. In Steels, L., Schreiber, G., and van de Velde, W., editors, *A Future of Knowledge Acquisition, Proc. 8th European Knowledge Acquisition Workshop (EKAW 94)*, pages 1–25. Springer.

[Schuster and Stuckenschmidt, 2001] Schuster, G. and Stuckenschmidt, H. (2001). Building shared ontologies for terminology integration. In Stumme, G., Maedche, A., and Staab, S., editors, *Proceedings of the KI-01 Workshop on Ontologies*. CEUR workshop proceedings volume 48.

[Selman and Kautz, 1996] Selman, B. and Kautz, H. (1996). Knowledge compilation and theory approximation. *Journal of the ACM*, 43(2):193–224.

[Sheth and Maes, 1993] Sheth, B. and Maes, P. (1993). Evolving agents for personalized information filtering. In *Proceedings of the ninth IEEE Conference on Artificial Intelligence for applications*, pages 345–352.

[Sowa, 2000] Sowa, J. (2000). Ontology, metadata, and semiotics. In Ganter, B. and Mineau, G., editors, *Conceptual Structures: Logical, Linguistic, and Computational Issues*, pages 55–81, Berlin. Springer-Verlag.

[Sowa, 1999] Sowa, J. F. (1999). *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Thomson Learning.

[Staab and Maedche, 2000] Staab, S. and Maedche, A. (2000). Ontology engineering beyond the modeling of concepts and relations. In *Proceedings of he ECAI'2000 Workshop on Applications of Ontologies and Problem-Solving Methods*, Berlin, Germany.

[Staab et al., 2001] Staab, S., Maedche, A., and Handschuh, S. (2001). An annotation framework for the semantic web. In *Proceedings of the First Workshop on Multimedia Annotation*, Tokyo, Japan.

[Stuckenschmidt, 2001] Stuckenschmidt, H. (2001). The buster reasoning module: A quick reference. Internal report of the buster project, Center for Computing Technologies, University of Bremen.

[Stuckenschmidt, 2002a] Stuckenschmidt, H. (2002a). Approximate information filtering on the semantic web. In *Proceedings of the 25th German Conference on Artificial Intelligence*, Lecture Notes in Artificial Intelligence. Springer.

[Stuckenschmidt, 2002b] Stuckenschmidt, H. (2002b). Approximate information filtering with multiple classification hierarchies. *International Journal on Computational Intelligence Applications*. Accepted for publication.

[Stuckenschmidt et al., 2000] Stuckenschmidt, H., Broekstra, J., Fensel, D., van Harmelen, F., Klein, M., and Horrocks, I. (2000). Catalogue integration - a case study in ontology-based semantic translation. FEW Report R-474, Vrije Universiteit Amsterdam.

[Stuckenschmidt and Euzenat, 2001] Stuckenschmidt, H. and Euzenat, J. (2001). Ontology language integration: A constructive approach. In *Proceedings of the Workshop on Applications of Description Logics at the Joint German and Austrian Conference on Artificial Intelligence*, Vienna, Austria.

[Stuckenschmidt et al., 2002] Stuckenschmidt, H., Hartmann, J., and van Harmelen, F. (2002). Learning structural classification rules for web-page categorization. In *Proceedings of Flairs'02, Special Track on the Semantic Web*. AAAI Press.

[Stuckenschmidt and van Harmelen, 2001a] Stuckenschmidt, H. and van Harmelen, F. (2001a). Knowledge-based validation, aggregation and visualization of meta-data: Analyzing a web-based information system. In Yao, Y., Zhong, N., Liu, J., and Ohsuga, S., editors, *Web Intelligence 2001*, number 2198 in Lecture Notes in Artificial Intelligence, pages 217–226. Springer.

[Stuckenschmidt and van Harmelen, 2001b] Stuckenschmidt, H. and van Harmelen, F. (2001b). Ontology-based metadata generation from semi-structured information. In *Proceedings of the first intenational conference on knowledge capture (K-CAP'01)*, pages 440–444. Sheridan Printing.

[Stuckenschmidt and Visser, 2000] Stuckenschmidt, H. and Visser, U. (2000). Semantic translation based on approximate re-classification. In *Proceedings*

of the Workshop on Semantic Approximation, Granularity and Vagueness at KR 2000.

[Stuckenschmidt and Wache, 2000] Stuckenschmidt, H. and Wache, H. (2000). Context modeling and transformation for semantic interoperability. In *Proceedings of the workshop 'Knowledge Representation meets Databases' KRDB-2000*, number 29 in CEUR Workshop proceedings, pages 115–126.

[Studer et al., 1998] Studer, R., Benjamins, V. R., and Fensel, D. (1998). Knowledge engineering: Principles and methods. *Data Knowledge Engineering*, 25(1-2):161–197.

[Templeton et al., 1987] Templeton, M., Brill, D., Dao, S., Lund, E., Ward, P., Chen, A., and MacGregor, R. (1987). Mermaid: a front end to distributed heterogeneous databases. *Proceedings of the IEEE*, 75(5):695–708.

[Thompson et al., 2000] Thompson, H., Beech, D., Maloney, M., and Mendelsohn, N. (2000). XML schema part 1: Structures. Working draft, W3C. http://www.w3.org/TR/2000/WD-xmlschema-1-20000225/.

[Turtle and Croft, 1991] Turtle, H. and Croft, W. (1991). Evaluation of inference network-based retrieval methods. *ACM Transactions on Information Systems*, 9(3):187–222.

[Uschold, 1996] Uschold, M. (1996). Building ontologies: Towards a unified methodology. In *16th Annual Conf. of the British Computer Society Specialist Group on Expert Systems*, Cambridge, UK.

[Uschold and Gruninger, 1996] Uschold, M. and Gruninger, M. (1996). Ontologies: Principles, methods and applications. *Knowledge Engineering Review*, 11(2):93–155.

[Valente et al., 1999] Valente, A., Russ, T., MacGrecor, R., and Swartout., W. (1999). Building and (re)using an ontology for air campaign planning. *IEEE Intelligent Systems*, 14(1):27–36.

[van Harmelen and Fensel, 1999] van Harmelen, F. and Fensel, D. (1999). Practical knowledge representation for the web. In Fensel, D., editor, *Proceedings of the IJCAI'99 Workshop on Intelligent Information Integration*.

[van Harmelen et al., 2001] van Harmelen, F., Patel-Schneider, P. F., and Horrocks, I. (2001). Reference description of the DAML+OIL (march 2001) ontology markup language. http://www.daml.org/2001/03/reference.html.

[van Harmelen and van der Meer, 1999] van Harmelen, F. and van der Meer, J. (1999). Webmaster: Knowledge-based verification of web-pages. In Ali, M. and Imam, I., editors, *Proceedings of IEA/AEI99*, LNAI, pages 147–166. Springer Verlang.

[van Heijst et al., 1997] van Heijst, G., Schreiber, A., and Wielinga, B. (1997). Using explicit ontologies for KBS development. *International Journal of Human-Computer Studies*, 46(2/3):183–292.

[van Rijsbergen, 1979] van Rijsbergen, C. (1979). *Information Retrieval*. Butterworths, London, second edition.

[Visser et al., 1997] Visser, P. R. S., Jones, D. M., Bench-Capon, T. J. M., and Shave, M. J. R. (1997). An analysis of ontological mismatches: Heterogeneity versus interoperability. In *AAAI 1997 Spring Symposium on Ontological Engineering*, Stanford, USA.

[Visser and Stuckenschmidt, 1999] Visser, U. and Stuckenschmidt, H. (1999). Intelligent, location-dependent acquisition and retrieval of environmental information. In Rumor, M., editor, *Information Technology in the Service of Local Government Planning and Management*. The Urban Data Management Society.

[Visser et al., 2002] Visser, U., Stuckenschmidt, H., Schuster, G., and Voegele, T. (2002). Ontologies for geographic information processing. *Computers in Geosciences*, 28:103–117.

[Visser et al., 2001] Visser, U., Stuckenschmidt, H., Wache, H., and Voegele, T. (2001). Using environmental information efficiently: Sharing data and information from heterogeneous sources. In Rautenstrauch, C., editor, *Environmental Information Systems in Industry and Administration*, pages 41–73. Idea Group.

[Voegele et al., 2000] Voegele, T., Stuckenschmidt, H., and Visser, U. (2000). Buisy - using brokered data objects for environmental information systems. In Tochtermann, K. and Riekert, W.-F., editors, *Hypermedia im Umweltschutz*, pages 68 – 73, Marburg. Metropolis Verlag.

[Wache, 1999] Wache, H. (1999). A rule-based mediator for the integration of heterogeneous sources (extended version). TZI–Berichte, University of Bremen, Bremen.

[Wache and Stuckenschmidt, 2001] Wache, H. and Stuckenschmidt, H. (2001). Practical context transformation for information system interoperability. In Akman, V., Bouquet, P., Thomason, R., and Young, R. A., editors, *Modeling and Using Context, Third international and Interdisciplinary Conference, CONTEXT 2001*, number 2116 in Lecture Notes in Artificial Intelligence, pages 367–381. Springer.

[Wache et al., 2001] Wache, H., Voegele, T., Visser, U., Stuckenschmidt, H., Schuster, G., and S.Huebner, H. N. (2001). Ontology-based integration of information - a survey of existing approaches. In [Gomez-Perez et al., 2001], pages 108–117.

[Weibel, 1999] Weibel, S. (1999). The state of the Dublin Core metadata initiative. *D-Lib Magazine*, 5(4).

[Wiederhold, 1992] Wiederhold, G. (1992). Mediators in the architecture of future information systems. *IEEE Computer*, March:38–49.

[Wiederhold, 1996] Wiederhold, G., editor (1996). *Intelligent Integration of Information*. Kluwer Academic Publishers, Boston MA.

[Williamson et al., 2000] Williamson, K., Healy, M., and Barker, R. (2000). Reuse of knowledge at the appropriate level of abstraction. In *Proceedings of the 6th International Conference on Software Reuse (ICSR'6)*, pages 58–73.

[Yarowsky, 1992] Yarowsky, D. (1992). Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. In *Proceedings of COLING-92*, pages 454–460, Nantes, France.

Samenvatting

Information sharing is het op een gemeenschappelijke manier toegankelijk maken van bestaande informatiebronnen, in plaats van het vergaren van nieuwe informatie en systemen om deze informatie te verwerken. Succesvolle information sharing heeft vele duidelijke voordelen, waaronder de volgende:

- Kwaliteitsverbetering van processen doordat meer en vaak completere informatie beschikbaar is.
- Kostenverlaging doordat informatiebronnen meerdere keren gebruikt kunnen worden en informatie niet opnieuw verworven hoeft te worden.
- Het voorkomen van het dubbel opslaan van informatie en van conflicten die hieruit voort kunnen komen.

Om efficiënte informatie sharing op te zetten moeten echter veel technische problemen worden opgegelost. Allereerst moet een geschikte informatiebron die gegevens voor een bepaalde taak bevat worden gelokaliseerd. Het vinden van geschikte bronnen is een probleem waarmee de vakgebieden *information retrieval* en *information filtering* zich bezighouden. Het verwerken en interpreteren van informatie uit verschillende bronnen brengt een andere probleem met zich mee: hoe kunnen de verschillende vormen van heterogeniteit van informatie worden opgelost:

Syntax: Het specifieke formaat waarin de informatie is gerepresenteerd kan verschillen. Dit probleem wordt voor het grootste deel opgelost door *wrapping technology* en het definiëren van standaarden zoals XML, die het gebruik van eigen formaten overbodig maken.

Structuur: De logische datamodellen van de informatiebronnen kunnen verschillen. In het database onderzoek zijn methoden voor schema integratie op basis van *views* ontwikkeld die in staat zijn om de meeste vormen van structurele heterogeniteit op te lossen.

Semantiek: De onderliggende conceptualisatie van de informatie kan verschillen. Het probleem van het integreren van informatie die is gebaseerd op verschillende conceptuele modellen wordt *context transformation* genoemd. Dit probleem wordt behandeld in sommige recente aanpakken op het gebied van *federated databases* en *data warehouses*.

Ontologieën worden gezien als een van de belangrijkste technologieën om heterogeniteit op het niveau van de semantiek op te lossen. De meeste van deze aanpakken zijn echter afhankelijk van het bestaan van goed gefundeerde data structuren die gebruikt kunnen worden voor het analyseren en uitwisselen van informatie. In dit proefschrift wordt behandeld hoe op ontologieën gebaseerde aanpakken kunnen worden gebruikt om semantische heterogeniteit op te lossen in semi-gestructureerde omgevingen, bijvoorbeeld op het World Wide Web. Om dit te bereiken moeten we oplossingen vinden voor de volgende problemen die samenhangen met het karakter van semi-gestructureerde omgevingen:

Ontbrekende Conceptuele Modellen: In semi-gestructureerde omgevingen hebben we geen toegang tot het conceptuele model van een informatiebron, noch tot het daaruit voortkomende logische data model. Dit gebrek aan structuur maakt het moeilijk om de context van de informatie te benoemen, wat noodzakelijk is om *context transformation* regels te definieren.

Onduidelijke Grenzen van het Systeem: In semi-gestructureerde omgevingen is het niet mogelijk om duidelijk te bepalen welke informatie relevant is omdat informatiebronnen vaak gewijzigd, verwijderd of toegevoegd worden. We kunnen daarom niet uitgaan van een vaste verzameling *context transformation* regels.

Heterogene Representaties: In een semi-gestructureerde omgeving kunnen we eveneens niet aannemen dat ontologieën op een uniforme manier zijn gerepresenteerd omdat er verschillende ontology representaties zijn voorgesteld. Dit betekent dat we ook op ontology-niveau moeten integreren.

Om deze problemen aan te pakken hebben we bijgedragen aan een raamwerk voor ontologie gebaseerde *information sharing* in semi-gestructureerde omgevingen, dat probeert een antwoord te formuleren op de volgende vragen:

- Hoe kunnen we ontologieën representeren en hierover redeneren in een semi-gestructureerde omgeving zonder een uniforme representatie?
- Wat voor soort ontologieën zijn nodig, hoe werken ze op elkaar in, en kunnen we het constructie process ondersteunen?
- Hoe kunnen we ontologieën en stukken informatie op een nauwkeurige en flexibele manier aan elkaar koppelen zodat dit bovendien nog in hoge mate geautomatiseerd kan worden?
- Hoe kunnen we ontologieën en logisch rederen over ontologieën gebruiken voor *nformation sharing* tussen web-gebaseerde informatiebronnen?

Dit proefschift, die bovenstaande vragen behandeld, is op de volgende manier gestructureerd:

Deel I: Semantische Interoperabiliteit

In het eerste deel van het proefschrift bespreken we de talen voor het representeren van expliciete modellen van de betekenis van informatie. Ook bespreken we het probleem van het bereiken van interoperabiliteit tussen deze talen, wat een voorwaarde is om deze modellen te kunnen gebruiken voor *information sharing*.

Er zijn een aantal talen voorgesteld om ontologieën op het World Wide Web te representeren. We beschrijven twee van deze talen — die hoogstwaarschijnlijk zullen worden geaccepteerd als standaard — in wat meer detail, namelijk RDF Schema en DAML+OIL. Verwijzend naar een recente overzichtsstudie van verschillende talen beargumenteren we dat ondersteuning voor de integratie van verschillende talen noodzakelijk is.

De huidige aanpak om interoperabiliteit mogelijk te maken is het definiëren van een standaardtaal waaraan iedereen zich moet houden. Wij beschrijven een nieuwe aanpak die flexibeler is dan betaande aanpakken maar die nog steeds bepaalde formele eigenschappen garandeert. We introduceren de algemene gedachte achter de aanpak en onderzoeken deze aanpak gedetailleerder voor het specifieke geval van terminologische talen — talen die relevant zijn voor het representeren van ontologieën. Onze aanpak is gebaseerd op een roosterstructuur van talen (een *semi-lattice*). Lokale transformaties van operatoren worden gebruikt om modellen die gerepresenteerd zijn in verschillende talen af te beelden op de taal die de laagste bovengrens in het rooster vormt. Daarbij beschouwen we transformaties die bepaalde formele eigenschappen bewaren. We beschrijven eveneens hoe deze aanpak met behulp van bestaande technologie geïmplementeerd kan worden.

Deel II: Ontologieën en Metadata

Het tweede deel van het proefschrift gaat over het probleem van het creëren van semantische beschrijvingen van informatie met de talen die in het eerste deel zijn behandeld. We bespreken verder hoe metadata kan worden gebruikt om voor het verbinden van ontologiën met de stukken informatie die ze beschrijven.

Op dit moment gaan de meeste systemen voor *information sharing* op het web uit van het bestaan van een globale ontologie. Nieuwe ontologieën moeten òf worden samengevoegd met bestaande ontologieën — zodat een nog grotere globale ontologie wordt gevormd — òf moeten worden gerelateerd aan alle reeds bestaande ontologieën. Beide aanpakken leveren problemen op: globale ontologieën zijn moeilijk te onderhouden en uit te breiden, het maken van afbeeldingen tussen lokale ontologieën is moeilijk en de inspanning die nodig is om deze afbeeldingen te creëren wordt groter voor iedere nieuwe ontologie.

Wij doen een voorstel voor een infrastructuur die het midden zoekt tussen globale en lokale ontologieën. Systemen kunen zo hun terminologie zelfstandig definieëren, maar de betekenis van termen kan nog steeds vergeleken worden met andere systemen zonder dat daar expliciete afbeeldingen tussen termen voor nodig is. Verder ontwikkelen en evalueren we een gespecialiseerde aanpak om deze infrastructuur te bouwen.

Een van de meest uitdagende problemen die samenhangen met het gebruik van ontologieën om het World Wide Web is het relateren van grote hoeveelheden informatie aan deze ontologieën. Dit probleem is al op verschillende manieren benaderd, bijvoorbeeld door het toepassen van *machine learning* technieken, of door het ontwikkelen van geavanceerde systemen om informatie te annoteren. Echter, de gebruiker moet nog steeds veel doen om de informatie aan de ontologieën te relateren. We presenteren een methode die verschillende methoden en technieken voor het bewerken, rederen en leren combineert zodat meta-data over de inhoud van informatiebronnen voor een groot gedeelte automatisch kan worden gegenereerd en gecontroleerd. Daarbij maken we gebruik van het weinige aan structuur dat beschikbaar is in informatie op het web.

Deel III: Information Sharing

In het derde deel van dit proefschrift behandelen we methoden om informatie te filteren en te integreren die gebruik maken van de expliciete modellen van de betekenis van informatie, zoals beschreven in de ontologieën en de metadata. Na een theoretische beschouwing over deze methoden beschrijven we een implementatie van deze methoden in een intelligent systeem voor *information sharing*.

Het werk rond het gebruik van ontologieën voor het filteren van informatie aan de ene kant, en redeneren over terminologieën aan de andere kant is vrijwel geheel gescheiden van elkaar in het huidige onderzoek rondom het semantic web. Redeneren over terminologieën wordt vooral tijdens de ontwikkeling van ontologieën toegepast om de consistentie te controleren en om verborgen specialisatie-relaties te vinden. De belangrijkste reden hiervoor is dat terminologische redeneermachines afhankelijk zijn van een compleet en homogeen model en alleen exacte gelijkenissen kunnen vinden op basis van logische bewijzen. Wij breiden de methoden voor redeneren over terminologieën op zo'n manier uit dat ze ook gebruikt kunnen worden voor het redeneren over heterogene ontologieën. We doen dit door exacte benaderingen van het bedoelde antwoord te berekenen. Dit overbrugt de kloof tussen het werk rondom het vertalen van betekenis van termen en redeneren over terminologieën.

Om de praktische toepasbaarheid aan te tonen van de ideeën en de methoden die in het kader van dit proefschrift zijn ontwikkeld, beschrijven we het BUSTER systeem, waarin veel van de ideeën geïmplementeerd zijn.

We beschrijven zowel de globale architectuur van het systeem als de rol van ontologieën en metadata in het systeem. We geven slechts een korte beschrijving van de redeneer module van BUSTER. In deze module zijn zowel onze aanpak voor het berekenen van de benaderingen van gelijkenis tussen concepten als de methoden voor het filteren van informatie met behulp van ontologieën geimplementeerd.

Conclusies

We hebben een aantal conlusies getrokken op basis van de resultaten die we hebben verkregen in ons onderzoek naar het ondersteunen van ontologiegebaseerde *information sharing* in semi-gestructureerde omgevingen. De conclusies zijn samengevat in de volgende principes:

Compatibiliteit: We moeten garanderen dat de representatie van ontologieën verenigbaar is met huidige standaarden voor het web. In ons geval moesten we garanderen dat de ontologieën in een op XML gebaseerde taal gerepresenteerd zijn dat we in staat zijn om ontologieën in DAML+OIL te verwerken. Zonder deze minimale eisen over homogeniteit zouden we problemen krijgen die redelijkerwijs niet oplosbaar zijn.

Sub-Optimaliteit: Redeneermethoden die werken met ontologieën in semi-gestructureerde omgevingen zijn sub-optimaal, dat wil zeggen, we kunnen er niet van uit gaan dat we in alle gevallen een definitief resultaat kunnen geven. Wanneer we te maken hebben met heterogeniteit tussen verschillende ontologieën moeten we onszelf beperken tot methoden die in de meeste gevallen een resultaat opleveren.

Lokaliteit: In semi-gestructureerde omgevingen kunnen we niet alleen vertrouwen op ontologieën die een *gemeenschappelijk* gezichtspunt modelleren. We zullen ook gebruik moeten maken van ontologieën om specifieke kennis over een informatiebron te representeren die niet gedeeld wordt door de andere bronnen. We hebben ontdekt dat dit lokale gezichtspunt op ontologieën het mogelijk maakt om te rederen over informatie. Bovendien helpt het ons om op een bottom-up manier een minimaal gemeenschappelijk model van een bepaald gebied te maken.

Zwakke Verbinding tussen informatie en ontologieën: Omdat er geen goed ontwikkeld datamodel is, zullen we moeten uitgaan van een erg zwakke verbinding tussen de ontologieën en de stukken informatie. We bereiken dit door complete web pagina's toe te wijzen aan ontologische categorieën, zonder de losse feiten op een pagina precies te beschrijven. Deze manier van verbinden bleek een goede keus, omdat de verbinding vrijwel geheel automatisch kan worden gemaakt en toch nog voldoende informatie biedt om informatie te filteren en vergelijken op basis van de betekenis.

SIKS Dissertatiereeks

====
1998
====

1998-1 Johan van den Akker (CWI)
DEGAS - An Active, Temporal Database of Autonomous Objects
1998-2 Floris Wiesman (UM)
Information Retrieval by Graphically Browsing Meta-Information
1998-3 Ans Steuten (TUD)
A Contribution to the Linguistic Analysis of Business
Conversations within the Language/Action Perspective
1998-4 Dennis Breuker (UM)
Memory versus Search in Games
1998-5 E.W.Oskamp (RUL)
Computerondersteuning bij Straftoemeting

====
1999
====

1999-1 Mark Sloof (VU)
Physiology of Quality Change Modelling; Automated modelling of
Quality Change of Agricultural Products
1999-2 Rob Potharst (EUR)
Classification using decision trees and neural nets
1999-3 Don Beal (UM)
The Nature of Minimax Search
1999-4 Jacques Penders (UM)
The practical Art of Moving Physical Objects
1999-5 Aldo de Moor (KUB)
Empowering Communities: A Method for the Legitimate User-Driven
Specification of Network Information Systems
1999-6 Niek J.E. Wijngaards (VU)
Re-design of compositional systems
1999-7 David Spelt (UT)
Verification support for object database design
1999-8 Jacques H.J. Lenting (UM)
Informed Gambling: Conception and Analysis of a Multi-Agent
Mechanism for Discrete Reallocation.

====
2000
====

2000-1 Frank Niessink (VU)
Perspectives on Improving Software Maintenance
2000-2 Koen Holtman (TUE)
Prototyping of CMS Storage Management
2000-3 Carolien M.T. Metselaar (UVA)
Sociaal-organisatorische gevolgen van kennistechnologie;
een procesbenadering en actorperspectief.
2000-4 Geert de Haan (VU)
ETAG, A Formal Model of Competence Knowledge for User
Interface Design
2000-5 Ruud van der Pol (UM)
Knowledge-based Query Formulation in Information Retrieval.
2000-6 Rogier van Eijk (UU)
Programming Languages for Agent Communication
2000-7 Niels Peek (UU)
Decision-theoretic Planning of Clinical Patient Management
2000-8 Veerle Coup (EUR)
Sensitivity Analyis of Decision-Theoretic Networks
2000-9 Florian Waas (CWI)
Principles of Probabilistic Query Optimization
2000-10 Niels Nes (CWI)
Image Database Management System Design Considerations,
Algorithms and Architecture
2000-11 Jonas Karlsson (CWI)
Scalable Distributed Data Structures for Database Management
====
2001
====

2001-1 Silja Renooij (UU)
Qualitative Approaches to Quantifying Probabilistic Networks
2001-2 Koen Hindriks (UU)
Agent Programming Languages: Programming with Mental Models
2001-3 Maarten van Someren (UvA)
Learning as problem solving
2001-4 Evgueni Smirnov (UM)
Conjunctive and Disjunctive Version Spaces with Instance-Based
Boundary Sets
2001-5 Jacco van Ossenbruggen (VU)
Processing Structured Hypermedia: A Matter of Style
2001-6 Martijn van Welie (VU)
Task-based User Interface Design
2001-7 Bastiaan Schonhage (VU)
Diva: Architectural Perspectives on Information Visualization
2001-8 Pascal van Eck (VU)
A Compositional Semantic Structure for Multi-Agent Systems Dynamics
2001-9 Pieter Jan 't Hoen (RUL)
Towards Distributed Development of Large Object-Oriented Models,
Views of Packages as Classes
2001-10 Maarten Sierhuis (UvA)
Modeling and Simulating Work Practice
BRAHMS: a multiagent modeling and simulation language for
work practice analysis and design

2001-11 Tom M. van Engers (VUA)
Knowledge Management:
The Role of Mental Models in Business Systems Design

====
2002
====

2002-01 Nico Lassing (VU)
Architecture-Level Modifiability Analysis
2002-02 Roelof van Zwol (UT)
Modelling and searching web-based document collections
2002-03 Henk Ernst Blok (UT)
Database Optimization Aspects for Information Retrieval
2002-04 Juan Roberto Castelo Valdueza (UU)
The Discrete Acyclic Digraph Markov Model in Data Mining
2002-05 Radu Serban (VU)
The Private Cyberspace Modeling Electronic
Environments inhabited by Privacy-concerned Agents
2002-06 Laurens Mommers (UL)
Applied legal epistemology; Building a knowledge-based ontology of
the legal domain
2002-07 Peter Boncz (CWI)
Monet: A Next-Generation DBMS Kernel For Query-Intensive
Applications
2002-08 Jaap Gordijn (VU)
Value Based Requirements Engineering: Exploring Innovative
E-Commerce Ideas
2002-09 Willem-Jan van den Heuvel(KUB)
Integrating Modern Business Applications with Objectified Legacy
Systems
2002-10 Brian Sheppard (UM)
Towards Perfect Play of Scrabble
2002-11 Wouter C.A. Wijngaards (VU)
Agent Based Modelling of Dynamics:
Biological and Organisational Applications
2002-12 Albrecht Schmidt (Uva)
Processing XML in Database Systems
2002-13 Hongjing Wu (TUE)
A Reference Architecture for Adaptive Hypermedia Applications
2002-14 Wieke de Vries (UU)
Agent Interaction: Abstract Approaches to Modelling,
Programming and Verifying Multi-Agent Systems
2002-15 Rik Eshuis (UT)
Semantics and Verification of UML Activity Diagrams for
Workflow Modelling
2002-16 Pieter van Langen (VU)
The Anatomy of Design: Foundations, Models and Applications
2002-17 Stefan Manegold (UVA)
Understanding, Modeling, and Improving Main-Memory
Database Performance